Design • Build • Program • Learn • Compete

SERVO

FOR THE ROBOT INNOVATOR

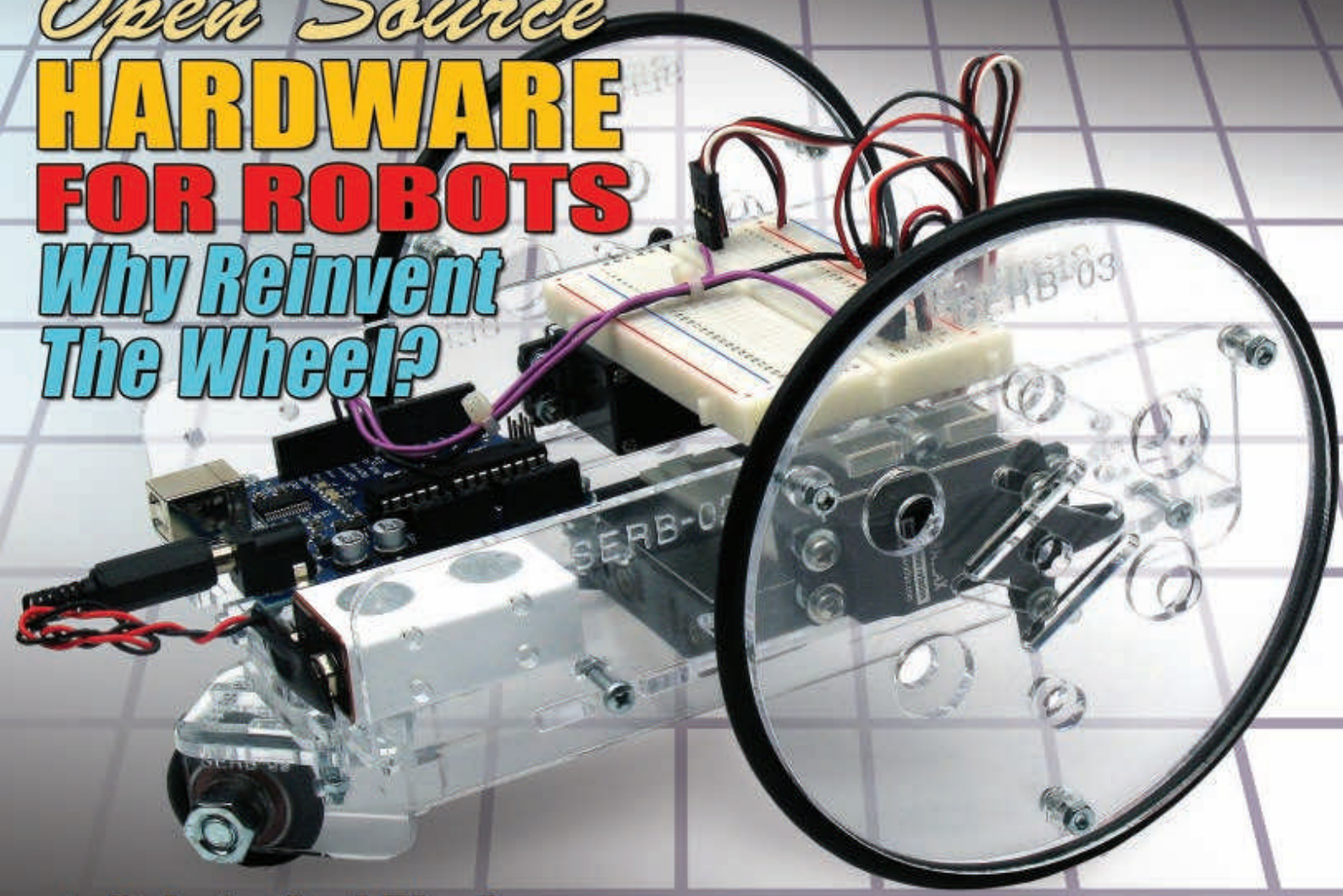
www.servomagazine.com

SERVO MAGAZINE

June 2013

Open Source HARDWARE FOR ROBOTS

Why Reinvent The Wheel?



- Robots And The Law
- 2013 VEX World Championships
- Designing JRover

Part Curiosity - Part Self Driving Car

Grab the Spotlight with Hitec!



Hitec's superb line of radios and servos are the clear choice for robot enthusiasts at the top of their game. Our flagship Aurora 9 computer radio with 2.4GHz Adaptive Frequency Hopping Spread Spectrum technology has unwavering reliability, endless programming capabilities and impressive, easy-to-master operation. The three-channel Aggressor SRX Pro pistol-grip radio with 10-model memory uses Direct Sequence Spread Spectrum technology to ensure its solid control signal. Choose one of these two or the Eclipse 7 Pro, Optic 6 Sport, Optic 5 or Aggressor 3DS. No matter what your project, Hitec has the gear to keep you and your bot in the spotlight!

HiTEC

12115 Paine Street • Poway, CA 92064 • 858.748.6948 • www.hitecrcd.com



Zumo Robot

Arduino-controllable tracked robot small enough for mini-sumo (less than 10 cm × 10 cm) and flexible enough for you to make it your own.



Maestro USB Servo Controllers

Compact, versatile servo controllers that offer USB, serial, and internal scripting control.



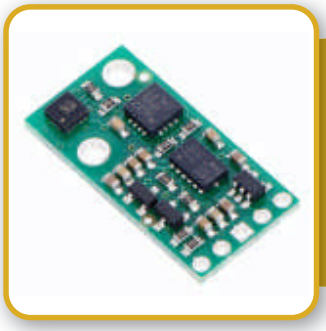
Jrk USB Motor Controllers

Highly configurable DC motor controllers that offer four control interfaces and can optionally be used with feedback for closed-loop speed or position control.



Wixel USB Wireless Module

General-purpose programmable module featuring a 2.4 GHz radio and USB: write your own software or load precompiled, open-source apps.



AltIMU-10 Gyro, Accelerometer, Compass, and Altimeter

Provides ten pressure, rotation, acceleration, and magnetic measurements that can be used to calculate absolute orientation and altitude.



Metal Gearmotors

Available in a wide range of motor types and gear ratios that let you choose the right combination of size, torque, and speed for your application.



Linear Actuators

Our 12 V linear actuators have dynamic load ratings ranging from 22 to 110 pounds and can be used in a variety of heavy-duty applications.



Custom Laser Cutting

With our custom part cutting service, you can quickly and economically realize intricate designs.

Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products — from actuators to wireless modules — that can help you take your robot from idea to reality.

Pololu
Robotics & Electronics

Find these products and more at **www.pololu.com**

SERVO MAGAZINE

06.2013

VOL. 11 NO. 6



PAGE 78

Columns

07 Robytes
by Jeff Eckert
Stimulating Robot Tidbits
- Definitely a Cool Robot
- Printable Bots on the Way
- Bot Surgery Lawsuits on the Rise
- A Mint for Your Tablet/Smartphone
- Robot Roadmap Available

10 Ask Mr. Roboto
by Dennis Clark
Your Problems Solved Here
Communicating with Dynamixel servos using MPIDE.

78 Then and Now
by Tom Carroll
Robots and the Law
What will the legal implications be if robots violate human rights?

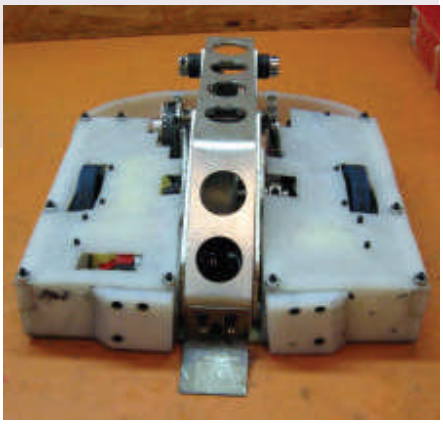
The Combat Zone...

28 BUILD REPORT:
The Quest for a Different Kind of Flipper: Improving the Trigger

31 *Then and Now:* A Decade Later With Brian Nave

34 EVENT REPORT:
FIRST Orlando Regionals: A New Set of Heroes Emerges

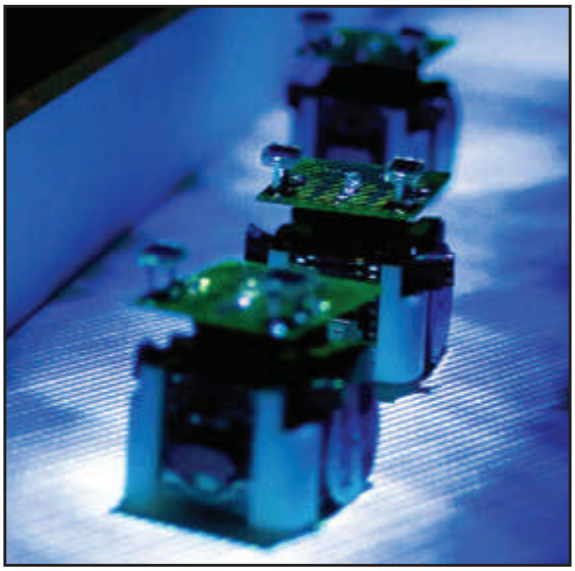
38 TIPS FROM THE PITS:
Speeding Up a Motor Change



Departments

06 Mind/Iron
15 New Products
17 Showcase
18 Events Calendar
58 SERVO Webstore
82 Robo-Links
82 Advertiser's Index

20 Bots in Brief
- Super Simple Swarm Slinkys
- Coman the Robotarian
- Robo Raven Takes Flight
- (Not So) Tiny Dancers
- ASK NAO
- Bionic Bugs
- Where's Hubo?
- A Nice Gesture
- Alice in Wanderland
- Turtles Times Two



SERVO Magazine **(ISSN 1546-0592/CDN Pub Agree#40702530)** is published monthly for $24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

In This Issue ...





PAGE 74

40 Open Source Hardware for Robot Projects

by R. Steven Rainwater

Did you know that there are community-developed hardware robot designs that are freely licensed and downloadable? If not, you need to check out this overview of different platforms that have been developed over the last two decades.

50 Interface and Communication Techniques to Control the World (or, at least H-bridges)

by Fred Eady

In the mechatronic world, it's all about control. See how to gain worldwide dominance using some simple CCS C code, a Wi-Fi module, and a handful of Microchip silicon.

62 Curiosity and the Self-Driving Car

by Paul Bouchier and Jason Henriksen

Preparing for the 2012 Roborama contest course inspired a melding of Mars rovers, self-driving cars, and lane-following technology that gave rise to JRover.

67 3D Printers Part 2: Assembly Highlights

by Michael Simpson

Follow the assembly process on four different 3D printers, so that next time we'll be ready to actually print a simple object.

74 The VEX 2013 World Championships

by R. Steven Rainwater

See highlights from this year's classic event.

Mind / Iron

by Bryan Bergeron, Editor

Telemedicine Robot Cleared for Launch

In addition to often fierce competition in the marketplace, medical device developers face the hurdle of obtaining clearance from the US Food and Drug Administration (FDA). The clearance process of proving effectiveness for certain medical conditions or settings can take years and a small fortune. The first autonomous remote presence robot to survive the federal gauntlet is the Remote Presence Virtual + Independent Telemedicine Assistant, or RP-VITA.

The RP-VITA — a combination of technologies from InTouch Health and iRobot (the maker of the Roomba vacuum cleaner) — is intended to allow a physician to support the care of a patient remotely. That is (unlike, say, the holographic doctor on Voyager which interacts directly with his patients), the RP-VITA supports a nurse or other healthcare professional in assessing a patient.

For example, the armless robot can't hold a stethoscope to a patient's chest, but it can direct a nurse to hold the stethoscope. The physician on the other end of the connection can listen to the sound and presumably make a diagnosis — even from the golf course.

Compared with the previous generation of InTouch medical robots – which I've used – this latest incarnation sports a sleeker body and a greatly improved user interface. There's an Apple iPad mounted front and center on the robot, meaning that any child over three or four can probably transform the robot into a Series 800 Terminator with a few finger presses. There's a camera and LCD panel mounted at head level that provides two-way visual communications.

Fundamentally, the technology is essentially that of a medical kiosk — the kind often found in malls and in larger companies — but on an intelligent, moveable platform. So, what's the big deal, from a practical perspective? Why don't nurses carry an iPad around and call in to the doctor whenever there's a problem? Or put everything on a cart?

I assume the folks at InTouch Health and iRobot would be quick to point out that their unit can be more easily shared by the staff in an ER. It's also difficult for a patient to stuff the robot into a pocket and walk out – definitely not the case with an iPad.

Technical features and quirks aside, the release of the RF-VITA illustrates what I believe is key to a disruptive technology. Notice that the robot isn't positioned as a nurse or doctor killer, destined to replace humans. Instead — as with the successful early expert systems and surgical robots — this robot is positioned as a tool to be used by clinicians. Such human-in-the-loop technologies are less likely to be sabotaged by those whose jobs are at risk.

Is this robot destined to become the delivery platform of choice for modern physicians? Probably not. But it does mark the start of an evolutionary change in the practice of medicine, and a foothold for other robotic platforms in the hospital, nursing home, and school clinic. Moreover, it's potentially a harbinger of a telepresence robot industry that enables virtual multitasking. One robot attends classes while another stands around a conference room table, discussing the merits of a new product launch.

Meanwhile, the operator, wearing a white dress shirt, tie, and plaid shorts, tries to look both studious and earnest, looking into his iPad camera while watching the waves crash into the beach. SV

FOR THE ROBOT INNOVATOR SERVO MAGAZINE

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

ASSOCIATE PUBLISHER/ VP OF SALES/MARKETING
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert
Tom Carroll
Dennis Clark
Michael Simpson
Paul Bouchier
Zac O'Donnell
Pete Smith
Jenn Eckert
Kevin Berry
R. Steven Rainwater
Fred Eady
Jason Henriksen
Collin Berry

CIRCULATION DEPARTMENT
subscribe@servomagazine.com

MARKETING COORDINATOR WEBSTORE
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

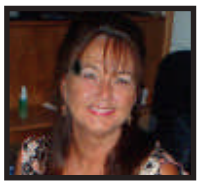
ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION
Sean Lemieux

Copyright 2013 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**

Printed in the USA on SFI & FSC stock.

Robytes

by Jeff and Jenn Eckert

Discuss this article in the *SERVO Magazine* forums at http://forum.servomagazine.com.

Definitely a Cool Robot



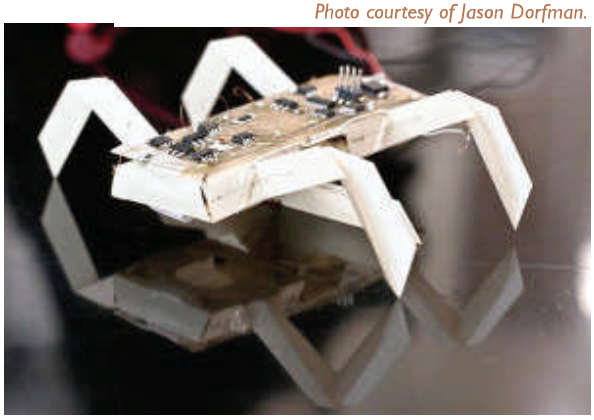
Yeti bot with the boom-mounted radar unit it is replacing.

With the Antarctic crawling with everything from marine biologists and geologists to astronomers to Sports Illustrated swimsuit models, it's getting to be a pretty popular place. Unfortunately, it's still one of the world's most hazardous locations, given the glaciers, crevasses, sea ice, mountains (Antarctica has the highest average elevation of all continents), winter temperatures that plunge down to -80°C (-112°F), and not a single Starbucks to be found. It's also an expensive place to visit. An average expedition will run you about $125,000, largely because of the costs involved in literally dragging people, equipment, fuel, and other supplies around using tractors. A big problem is that these vehicles are in constant danger of overloading a snow bridge and plunging into a pit, leaving them buried as much as 200 ft below the surface. In the past, tractors have employed ground-penetrating radar mounted on booms to detect weak spots, but this method provides only an iffy 2.5 second warning before you fall in.

Recently, the National Science Foundation (**www.nsf.gov**) got together with some students from Dartmouth and an Army Engineering lab, and came up with a solution. The "Yeti" robotic rover is now being deployed to head up the convoys and provide much earlier detection. The four-wheeled Yeti's radar generates a continuous waveform of subsurface layers to distinguish between solid ice and snow bridges, and since it weighs only 150 lb (68 kg) it isn't as likely to drop through. (That's a relatively cheap $25,000 price per unit, so apparently, it's not a huge loss if it does.) NSF estimates that Yeti-guided trips to McMurdo Station alone with save $200 million per year.

Printable Bots on the Way

Photo courtesy of Jason Dorfman.



Insect-like robot designed and printed using the new process.

Providing yet another reason why 3D printers are likely to be must-have computer peripherals in coming years, the MIT Computer Science and Artificial Intelligence Lab (CSAIL, **www.csail.mit.edu**) is heading up a project to "reinvent how robots are designed and produced." The aim is "to develop a desktop technology that would make it possible for the average person to design, customize, and print a specialized robot in a matter of hours." The five-year project is backed by a $10 million NSF grant, and also involves team members from the University of Pennsylvania and Harvard. According to MIT's Prof. Daniela Rus, "Our vision is to develop an end-to-end process; specifically, a compiler for building physical machines that starts with a high level of specification of function, and delivers a programmable machine for that function using simple printing processes."

When the platform is available, we should be able to simply identify a household problem, choose an off-the-shelf CAD design, customize it to suit our needs, and have a printed, assembled, programmed, and operational bot running within 24 hours. For a video of the pictured insect bot, just search "printable robots" on YouTube.

Bot Surgery Lawsuits on the Rise

If your local hospital has signed you up for surgery on one of the $1.5 million da Vinci robotic systems, you might first want to determine how much experience the surgeon has with the device. According to a Bloomberg news item, at least 10 lawsuits have been filed in recent months claiming that patients have been injured by robotic procedures, mainly as a result of poor training provided by Intuitive Surgical, Inc. — maker of the nearly 1,400 systems that are currently used in US hospitals. Intuitive chief medical advisor Myriam Curet described the training as "quite extensive," but all we're talking about is an online quiz, practice simulations, and a one-day onsite training course. It is also recommended that new doctors observe at least one operation, perform dry runs, and be assisted by an experienced surgeon for the first few procedures. Ultimately, "We cannot require anything," Curet noted, so it is up to the doctors to determine how much training they need.



Pay no attention to that man behind the console. ©Intuitive Surgical, Inc.

Mayo clinic researchers have determined that it takes 90 operations to become proficient in gynecological surgery, and a spokesman for the David Geffen School of Medicine noted that it can take more than 200 rounds of prostate surgery for a doctor to get it down. One result of the controversy is that the AAGL — formerly known as the American Association of Gynecologic Laparoscopists — is working on recommendations that will probably include simulator training similar to what airline pilots receive. "In the same way that airline pilots are credentialed to fly 787s, we want to make sure that before surgeons get in the cockpit that they have seen all scenarios they could possibly see," noted a surgeon who regularly uses the machines. In the meantime, at least make sure that the guy working the console is actually wearing scrubs, not a janitorial uniform.



The Auto Mee S robotic screen cleaner.

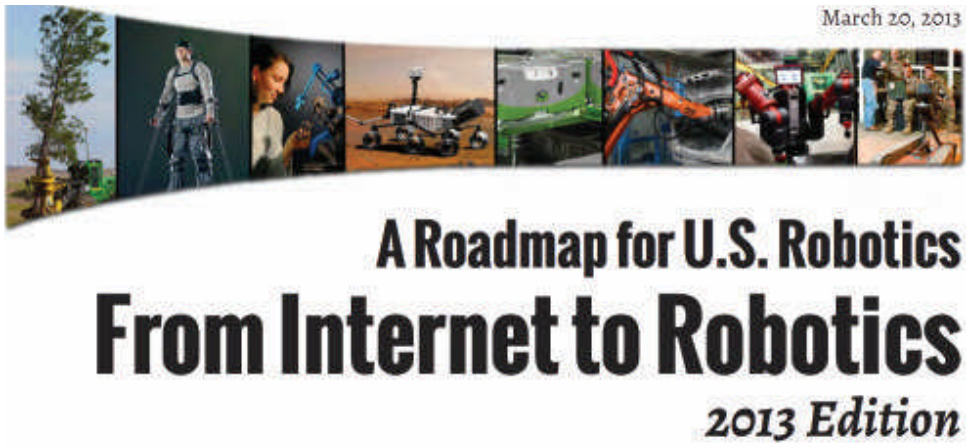
A Mint for Your Tablet/Smartphone

If you've been suffering from exhaustion, tennis elbow, cramps, backaches, or other maladies caused by the drudgery of cleaning the display on your tablet and/or smartphone, rejoice! Your agony is coming to an end, thanks to Japan's Takara Tomy Corporation (**www.takaratomy.co.jp**). For much less than the cost of a da Vinci, you can pick up the Auto Mee S robotic vacuum to do the job for you. Consider it as basically a mini version of iRobot's Mint floor sweeper.

Just plop it onto your mobile device's screen, and in a blink of an eye (assuming it takes you up to eight minutes to blink) it will be shining like the rising sun over Tokyo. The official list price has been reported at $17, but as of this writing you can order one from Amazon for only $33. Batteries, of course, are not included. Such a deal!

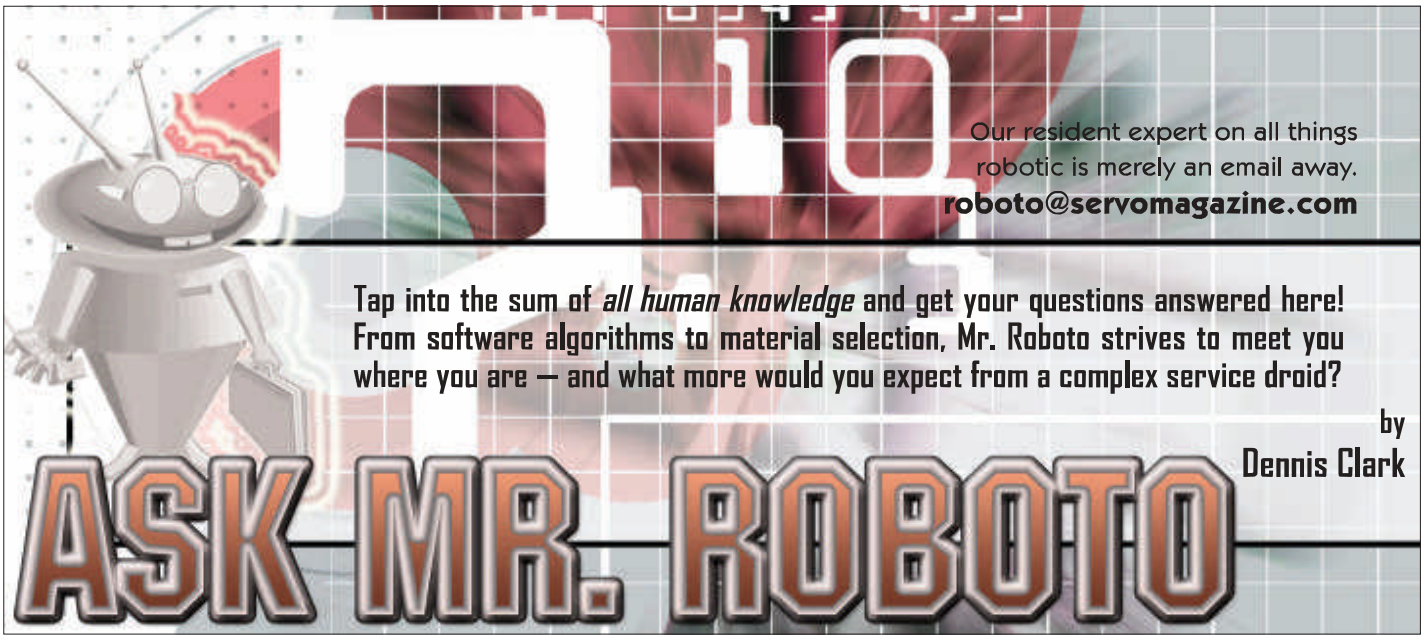
Robot Roadmap Available

In case you didn't realize it, there is a Congressional Robotics Caucus (CRC, **www.roboticscaucus.org**) that was formed in 2007 "to focus on key issues facing the nation's robotics industry and related emerging technology." In its list of goals, the caucus observes, "Robots go where it's dirty, dull, or dangerous," so it's obvious that they must be right at home in the halls of Congress. Every year, the CRC compiles its Roadmap for US Robotics with the assistance of Georgia Tech, Carnegie Mellon, and a slew of other industry experts, and prints it up for whoever is interested. If that includes you, you can download the 137 page 2013 edition by aiming your browser at **www.jkeckert.com/freedownloads/2013RoboticsRoadmap.pdf**. SV



The 2013 Robotics Roadmap — 137 pp of everything you need to know.





Our resident expert on all things robotic is merely an email away.
roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

ASK MR. ROBOTO

by
Dennis Clark

Summer is here, and now instead of robots competing for time with the need to go skiing, robots are competing with the need to go biking, hiking, and other stuff. What I need to do is get involved with some outdoor robot activities. Perhaps Robo-Magellan or something like that. It could satisfy the need to be outdoors while scratching the robot itch. So many decisions to make! Anyway, time to get going on the question that I spent many hours solving this month. Onward!

Q. I want to control my Bioloid Dynamixel servos with a Digilent UNO32 or Max32 controller on a new robot. How can I talk to these servos with MPIDE? I read that it is a totally different type of servo than a hobby servo. Can you help me?

— Kevin

A. That is a good question! You are correct when you say that the communications protocol is different from a hobby servo — *way* different from a hobby servo. My first step in answering this question is to find a document discussing exactly how one communicates to the Dynamixel servo. I found the most thorough document discussing the AX12a at **www.agaverobotics.com/products/servos/robotis/ax12/docs/AX-12.pdf**.

Trossen Robotics has a similar but not as technical document at **http://support.robotis.com/en/techsupport_eng.htm#product/dynamixel/ax_series/dxl_ax_actuator.htm**. Pointers to user open sources can be found at **www.agaverobotics.com/products/servos/#?node=products**.

I have a three year old Bioloid Comprehensive kit, so I chose it to work with the AX12a servo for this article. The first thing we need to do is find out how to "talk" to the Dynamixel servo bus. The Dynamixel communications channel is a bus structure; by *bus* I mean that a single set of wires can talk to every device on the bus. Hobby servos — by comparison — have a set of wires for *each* servo individually attached to a controller. The bus configuration uses a LOT few wires.

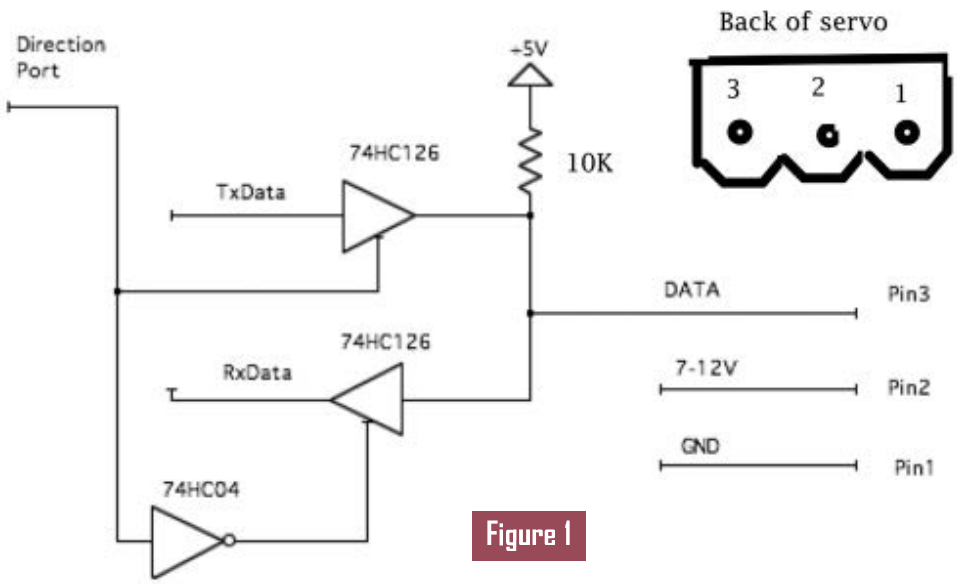


Figure 1

Part 1: The Hardware

Figure 1 shows what

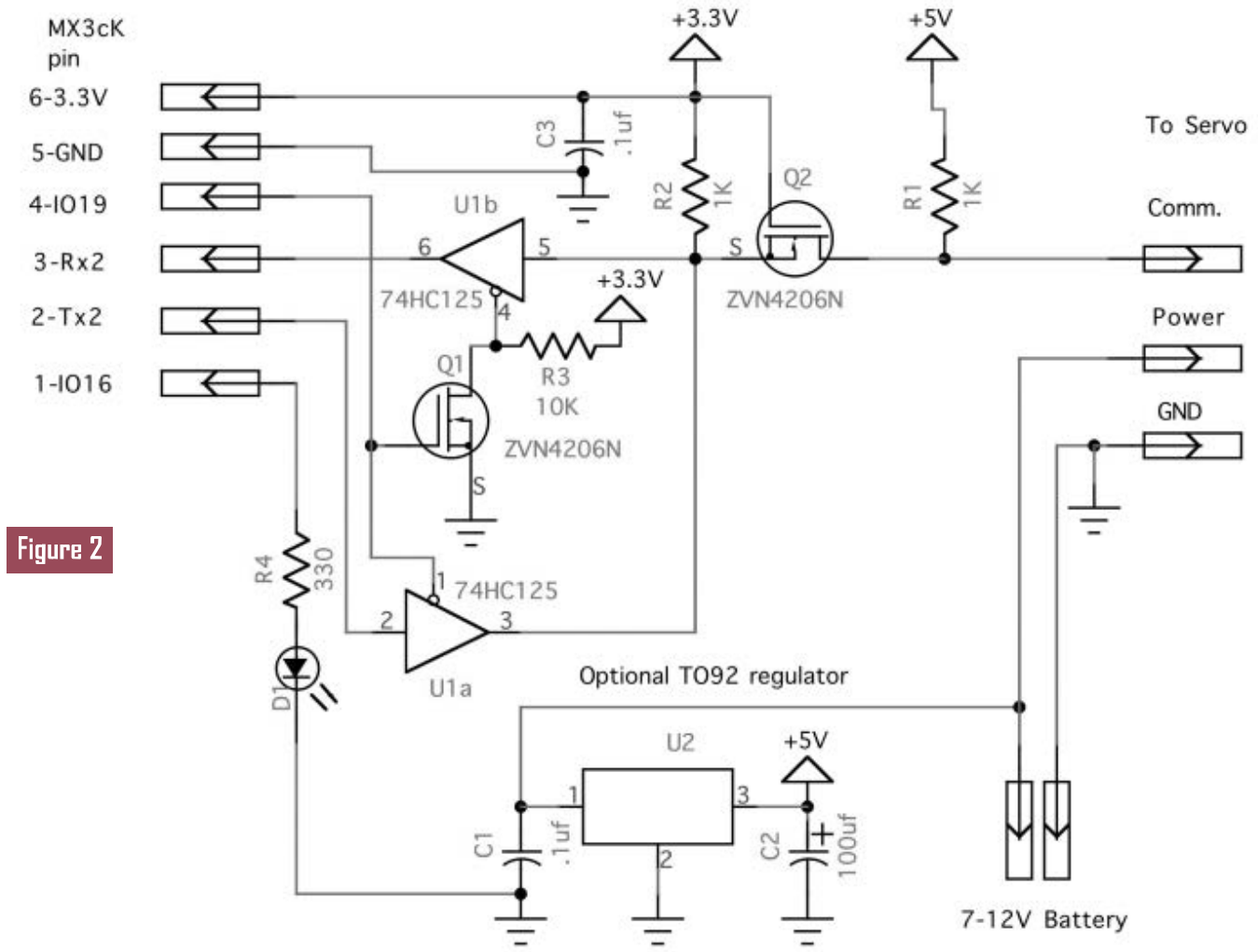


Figure 2

Robotis (makers of the Bioloid) used in their CM-5 robot controller to talk to the servos. I am not much of an artist, so my cable socket drawing in the upper right of the figure isn't all that pretty.

The Dynamixel bus is what is called *half-duplex*. This means that only one device can talk at a time. In this case, the *Master* (CM-5) talks to a servo, and a servo *Slave* then responds. According to the documentation, the data is transmitted using a UART asynchronous serial format, which is good for us.

The default data rate is 1 Mbps (one million bits per second). That is *really* fast, which means we need to be careful in our hardware design — which is our first problem.

There is another problem we need to deal with: The Dynamixel bus is 5V. The UNO32 and Cerebot (MX3cK) are 3.3V. (Rats, another voltage level converter problem!)

Well, we've dealt with the voltage converter problem before; using an N-channel FET, we get bi-directional voltage level conversion. Both Adafruit and SparkFun provide FET-based logic level conversion at:

Adafruit: **http://adafruit.com/products/757#Description**

SparkFun: **www.sparkfun.com/products/8745**

My own design uses through hole components, since that is what I have in my junkbox. Like these two designs, 10K resistor pull-ups are on the 5V and 3.3V sides. That takes care of part 2, but when I tested the design on my scope (when it didn't work), I saw that there were serious capacitive slews on the data line.

A slew occurs when a signal is changing so fast that the load getting the signal acts like an RC charging circuit, then that capacitor in the RC circuit can't charge fast enough for the signal to be properly seen. Well, we can't change the 'C' part of the circuit, but we sure can change the 'R' part!

My converter circuit uses 1K resistor pull-ups which reduces the resistance of the circuit 10X. Further testing with this change showed an acceptable slew rate and voila! My circuit started to work.

Figure 2 is the complete circuit design. I did not have any 74HC126s lying around, so I used a 74HC125 which is the same, but the enable on the buffers is active low instead of active high.

I wanted to keep my design compact so that I could plug it into a *PMOD* port on the MX3cK, so I

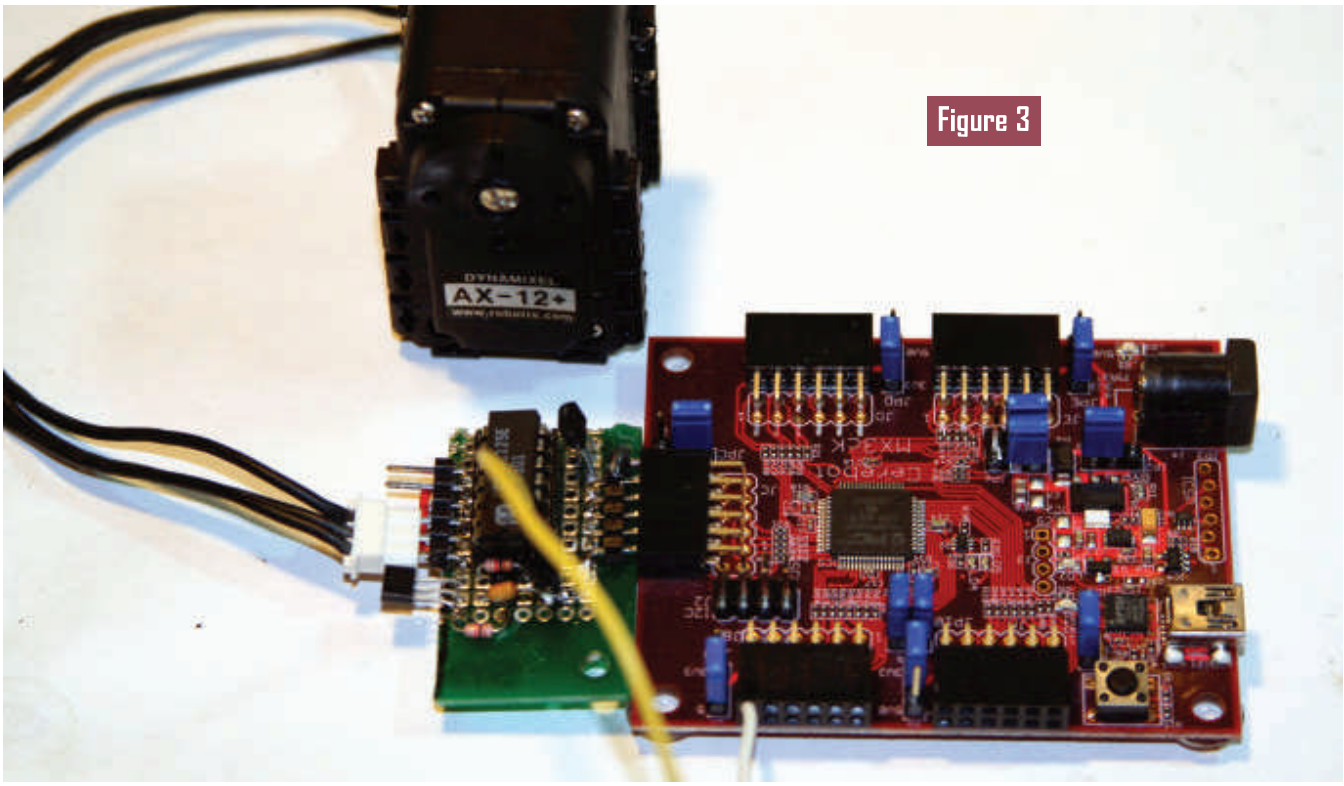


Figure 3

didn't use a 74HC04 inverter. I just used another one of my N-channel FETs to handle the circuit that switches between transmit and receive.

The 5V regulator part of the circuit is there should you want to use an UNO32, which won't have a 5V power option on a port. The MX3cK, however, does have the option of selecting 5V or 3.3V to power an attached circuit board. So, I used 5V from another port connector to give me the 5V I needed for the *DATA* line pull-up to the AX12a servo.

Because I could — and I think that LEDs are useful — and I had an extra I/O pin on the *PMOD* connector, I put an LED there to be blinked for any reason I wanted.

Figure 3 shows my assembled *PMOD* expansion board connected to the MX3cK and an AX12a servo plugged into it. You can use any battery from 7V to 12V with these servos. Be careful that you connect the servo correctly — I didn't have a polarized connector to use for doing this, and found out that the AX12a is a very robust servo that has been reverse-plug-protected. (Whew!) Maybe I just got lucky. Either way, note how the servo needs to be connected and do it correctly.

In electrical engineering, there is a concept with logic circuits called *fanout*. This is basically how many gates a single gate can *drive*, or talk to. I have not done careful experimenting or datasheet analysis of how many AX12as my circuit can talk to before the driver on my board is over-stressed and the servos don't get good data. I know that at least three are fine. I'll be doing more work with this and will report my findings.

Part 2: The Software

Now that the hardware is done and working, we need software that will drive this bus at 1 Mbps. This is a LOT faster than most serial connections. Each bit will take 1 µs to transmit, and each byte then takes 10 µs (eight bits of data; one stop and one start bit.) This means that you can send a lot of commands very quickly.

This is essential if you are going to control a robot with 16 or more degrees of freedom. Timing becomes more critical, however, so we need to be careful about when we are transmitting and when we expect to

Listing 1: Setup function.

```
void setup() {
  uint8_t n;

  pinMode(nTxU2, OUTPUT);    //~Tx/Rx: low transmit, high receive
  pinMode(DLED, OUTPUT);     // Das Blinken Lights, or LED

  Serial1.begin(1000000);    // Talk to Dynamixel AX12a at 1Mbps
  Serial.begin(115200);      // Talk to computer at 115.2Kbps

  LookBus(20);               // Look for servo, address 0-20

  Serial.println("Ready to roll\n");
}
```

Listing 2: Data setups and shortcuts.

```
#define DLED 16
#define nTxU2 19

// Dynamixel commands
#define DM_PING        0x01
#define DM_READ_DATA   0x02
#define DM_WRITE_DATA  0x03
#define DM_REG_WRITE   0x04
#define DM_ACTION      0x05
#define DM_RESET       0x06
#define DM_SYNC_WRITE 0x83

// Control Table memory addresses
uint8_t  ct[25] = {3,4,5,6,8,11,12,
13,14,16,17,18,19,24,25,26,27,28,29,30,32,34,4
4,47,48};
// Control Table memory bytes, 1 byte is a
// single value, 2 bytes means a 16 bit
// word
uint8_t  cb[25] = {1,1,1,2,2,1,1,1,2,1,1
,1,1,1,1,1,1,1,2,2,2,1,1,2};

// Command Table IDs
typedef enum DMControls_e
{
  dmID =0,
  dmBaud,
  dmRetDelay,
  dmCWAngleLimit,
  dmCCWAngleLimit,
  dmHighTempLimit,
  dmLowVoltLimit,
  dmHighVoltLimit,
  dmMaxTorque,
  dmStatRetLevel,
  dmAlarmLED,
  dmAlarmShutdn,
  dmReserved,
  dmTorqueEn,
  dmLED,
  dmCWCompMargin,
  dmCCWCompMargin,
  dmCWCompSlope,
  dmCCWCompSlope,
  dmPosition,
  dmMovSpeed,
  dmTorqueLimit,
  dmRegInst,
  dmLock,
  dmPunch
} DMcontrols_e;

uint8_t cmd[20];      // The protocol takes
                      // several bytes per
                      // command
uint8_t cmdLen;
uint8_t stat[20];
uint8_t statLen;
uint8_t err;
```

get a reception from the servo.

It is true that to control the servo we only need to write — and we can ignore the response packets from the servo — but if we ignore those received data packets at the wrong time, we will collide with our command transmissions and odd things will occur.

I am not going to repeat the descriptions of the commands and status packets and values here; you can get those from the first document link that I gave previously.

The driver that I created is not very sophisticated since I didn't have a lot of time. It is a simple set of functions, however, that can be used as the core of a library to control Dynamixel servos. Feel free to innovate!

Firstly, you'll notice that I'm using *uint8_t* and *uint16_t* as variable defines. These are GNU open source standard defines and work great to tell the coder eight-bit and 16-bit data values. Using *char* and *int* does not impart that much information.

The first part of the code to discuss is the setup for everything.

Listing 1 shows the MPIDE (just like Arduino) *Setup()* function:

The *LookBus()* function will scan the Dynamixel bus looking for a servo at address 0 through 20. It will print out only the status packets received from servos that respond.

This is the really simple part. **Listing 2** shows the data definitions that I made global to simplify the code. (Yes, globals are evil, but for limited memory embedded systems, they are a *good* thing!)

The data arrays detail the *Control Table* addresses and parameters that a *DATA WRITE* or *DATA READ* command use to change or read the current servo settings. That huge *enum* is a simple way to specify what element in the arrays to use for the commands.

As you read through the code, you'll see how I use them. It really does make it easier than remembering a bunch of magic numbers that do things.

I have made several functions that handle specific commands: *CmdPing()*, *CmdLED()*, *CmdMove()*, *CmdSpeed()*, and *CmdReadControl()*. The functions of these commands are suggested in their names: ping for a servo; turn the LED on or off; move the servo to a specific goal; set the servo transit speed; and read from the control table.

These functions will format the *cmd[]* byte array for a command transmission. A command packet requires a *checksum* at the end of it to handle simple detection of transmission errors.

The function *GetCSUM()* does this checksum. When the command is properly formatted, the code calls the *SendCmd()* function. This function is at the heart of communicating with the Dynamixel bus. **Listing 3** is this function.

There are lots of details to discuss here. I use the standard Arduino (MPIDE) *Serial1.write(array, arrayLen)* function call that transmits un-translated binary data out the second UART port after I set the *~Tx/Rx* bit low.

The MPIDE serial library doesn't have a call to tell me when the transmission is complete, but the PIC32 UART status register does —

that is what the *U2STAbits.TRMT* bit tells us. We don't want to flip the circuit to receive data while we are still transmitting, so we look for the transmission to be complete.

When complete, we flip the *~Tx/Rx* line high to receive. I have found that the act of flipping from Tx to Rx causes a false UART start bit to be seen which always gave me an extra 0xFF at the start of the status packet. My (less than perfect) solution at the moment is to immediately flush the UART receive buffer by reading a byte.

The next section looks for the start of a status response packet, which is two 0xFF bytes. If it doesn't get the first of these bytes within 500 µs, we will time out and return an error token, 0xFF. The function then starts another loop that gets the rest of the status packet. It too will time out if this phase doesn't complete within 500 µs.

In reality, the majority of the status packets received will be under 10 bytes — which at 10 µs per byte is under 100 µs. This is a comfortably conservative timeout. Best of all, it seems to work great with the 1-3 servo busses that I've tested it with.

The default AX12a response delay is 160 µs, but this is a configurable control table parameter, so you can't count on that. This is why I use this method of watching and timing out to wait, rather than a timer. It will work no matter what the response delay is set to be. The MX3cK is clocked at 80 MHz. That is *plenty* fast to respond to this serial protocol.

The rest of the code in this file just sends and receives data for various commands to cause things to happen on the servo. The AX12a has a user settable LED and tons of settings and things to read back on it. I'm looking forward to playing more with this very powerful setup!

The MPIDE project for this article is located at the article link as DynamixelDriver.zip. Feel free to expand on it and please, let me know what you do. (I'd love to know!) In my case, I'm planning to replace the CM-5 controller on my Bioloid with an MX3cK board and a custom LiPo charger.

Well, that's it for another Mr. Roboto. Time flies so fast when you are having fun. However, when you are blocked and not having fun, please send those cards, letters, and emails to me at at roboto@servomagazine.com and I'll do my best to answer them.

Until next month, keep on building robots! **SV**

Listing 3: SendCmd function.

```
uint8_t SendCmd(uint8_t *cmd, uint8_t cLen, uint8_t *stat, uint8_t
*sLen)
/*
*   Send Dynamixel command, return status in the cmd array
*   INPUT: cmd[] to send , cLen is length of command
*   OUPUT: stat[] status returned , sLen is length of status
*
*   RETURNS: non-zero if an error
*/
{
  uint8_t n = 0;
  uint32_t t;

  digitalWrite(nTxU2, LOW);          // writing out cmd
  Serial1.write(cmd, cLen);

  while(U2STAbits.TRMT == 0);        // Wait for all bytes to go out
  digitalWrite(nTxU2, HIGH);         // time to read back status
  Serial1.read();                    // switch creates false start bit
  t = micros() + 500;
  while (micros() < t)               // wait for reception to start
  {
    if (Serial1.available())
    {
      stat[0] = Serial1.read();
      if (stat[0] = 0xFF)
      {
        n++;
        break;
      }
    }
  }

  if (n == 0)                        // never got start byte
  {
    return(0xFF);                    // error flag
    *sLen = 0;
  }

  t = micros() + 500;
  while(micros() < t)                // Get the rest of the packet
  {
      while(Serial1.available() > 0)
      {
        stat[n++] = Serial1.read();
      }
  }
  *sLen = n;                         // number of bytes in the status

  return(stat[4]);                   // return the error byte
}
```

NEW PRODUCTS



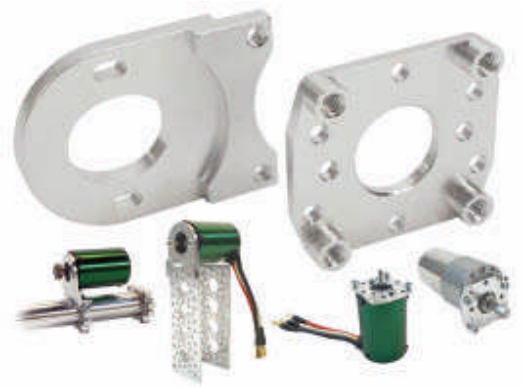
Aluminum Channel and Brackets

ServoCity now offers a new line of Actobotics™ aluminum channel and brackets. The unique overlapping hole patterns (0.770 and 1.50) found throughout the Actobotics product line allow for virtually unlimited mounting possibilities. The 6061-T6 aluminum construction offers durability and strength, while remaining light weight. The channel is offered in various lengths ranging from 1.5 inches to two feet long. Each channel is extruded during the manufacturing process, offering greater precision than bent aluminum. The pattern utilizes a 0.50" diameter center hole that is compatible with many different sizes and styles of ball bearing mounts (also available at ServoCity) which make it possible to incorporate precision shafting and tubing into your assembly. The Actobotics aluminum brackets add versatility to the channel by providing nearly endless design and attachment options. From single flat brackets, to dual-side angled brackets, these components make it simple to attach channel to channel, or channel to other components.



Aluminum R/C Motor Mounts

The all new Actobotics™ motor mounts from ServoCity allow you to easily integrate gear motors and R/C car motors into robotic projects. The mounts are designed to work with Actobotics channel and brackets. Constructed from 6061-T6 aluminum for superior strength, these mounts won't flex or bow. The multiple motor mounting holes are compatible with the majority of motors and gearmotors commonly found in both the R/C car and robotic markets. Motor Mount B (#555128) places the motor shaft directly in line with the center of the hole pattern found throughout Actobotics components. Motor mount C (#555124) is an offset style mount that steps the motor to the side of its mounting surface. When used in conjunction with other Actobotics components, this mount maintains the 1.5" center-center hole spacing between the shaft and the hub pattern on the adjacent component. Users can then choose from a number of gear, pulley, or belt ratios to ensure proper gear mesh or belt tension. The motor mounting holes are slotted for easy fine-tuning of the gear mesh or motor alignment. Both motor mounts utilize a 1.5" hole pattern which is found throughout the Actobotics line. There are 3 mm screws included to attach a motor or gearmotor. Pricing for Motor Mounts B and C is $4.99.



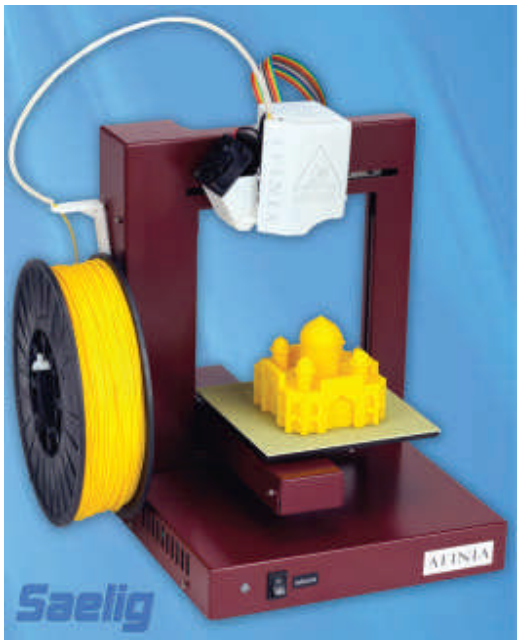
For further information, please contact:

ServoCity Website: **www.servocity.com**

Ready-to-Run 3D Printer

Saelig Company, Inc. is now offering the award-winning Afinia H479 rapid-prototyping 3D printer — a self-contained unit that provides an "immediate availability" 3D prototyping experience. The H479 3D Printer comes fully assembled — with easy to install software for both the PC and Mac — and can prototype a part up to five inches in each dimension; accurate to within .2 mm (.008"). A heated build platform prevents model warping during printing.

The 3D printing software supplied with the printer features an easy-to-use interface for laying out, orienting, duplicating, and scaling designs, importing standard STL files from design software such as SolidWorks and others. Using inexpensive, high quality ABS filament in a wide array of colors, this 11lb portable 3D printer can print in a stand-alone mode, disconnected from a PC. The software



auto-generates support material to ensure stable printing of arcs and overhangs

The compact 10" x 10" x 14" H479 3D Printer is small enough for any desktop, and ready for instant use. It is ideal for engineers, educators, and individual experimenters. Made in the USA by Afinia, the FCC-approved printer is available for $1,499. It is supplied with easy-to-use software, handling tools, and a starter spool of 1.5 lbs ABS plastic filament.

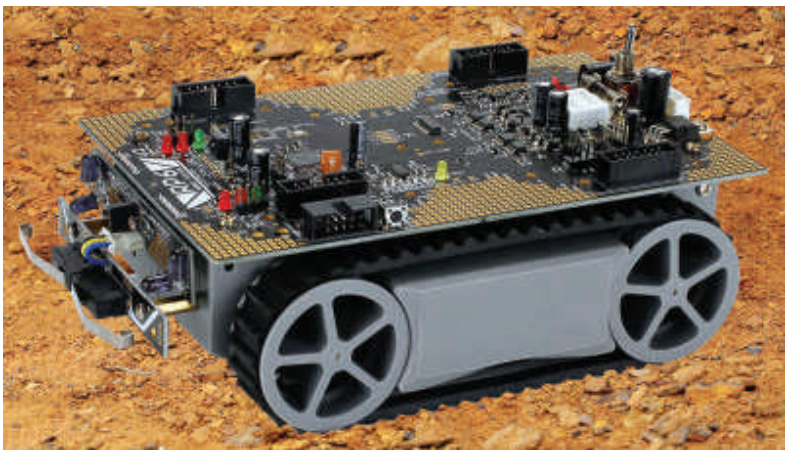
For further information, please contact:

Saelig Website: **www.saelig.com**

RP6v2 C-Programmable Robotic Vehicle

Global Specialties has introduced the RP6v2 C-Programmable Robotic Vehicle, available for $199. The RP6v2 is an economical autonomous mobile robot system which provides an introduction to the world of robotics. It is designed for beginners, as well as experienced electronics and software developers. Programmable in C, the RP6v2 has many possibilities for expansion as programming skills grow. The RP6v2 is ideal for educational curriculum at universities, trade schools, high schools, and hobby users. With an extensive manual, lots of example programs, and a huge C function library, programming is simple and users can instantly start experimenting with the robot. All library and example programs are open source (GNU GPL).The RP6v2 features include:

- ATmega32 eight-bit RISC microcontroller with 8 MIPS and 8 MHz clock.
- Delivered fully assembled (no soldering needed).
- CD with software, 138 page manual, and many extras.
- AVR-GCC and RobotLoader open source software for



MATLAB Support Package for Analog Discovery Design Kit

Digilent, Inc., announces the availability of a MATLAB® support package for their Analog Discovery hardware. With this support package available as part of MATLAB R2013a, customers may now test their circuit performance with Analog Discovery and bring that data to MATLAB for numerical computation, visualization, and programming. Together, MATLAB and Analog Discovery will give users the ability to generate, manipulate, and interactively utilize their collected data. The MATLAB support package allows professors, students, and researchers to perform a variety of tasks in MATLAB, including:

1. Read data from the two oscilloscope channels (analog input).
2. Control and generate data from the two waveform generators (analog output).
3. Characterize ICs and measure circuit and IC component behavior.
4. Configure the sampling rate of the Analog Discovery device.
5. Trigger the start of data acquisition.
6. Find and display Analog Discovery device settings.

The Analog Discovery – a USB powered test and measurement device – combines a dual-channel oscilloscope, a two-channel waveform generator, a 16-channel logic analyzer, and many other instruments into a USB-powered, low cost device.

MATLAB is a high-level language and interactive environment for numerical computation, visualization, and programming. With MATLAB, users can acquire and analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions offer the availability to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java™. The support package is free for users that already have the MATLAB R2013a. The Analog Discovery itself is $199. For further information, please contact:

Digilent, Inc. Website: **www.digilentinc.com**

use with Windows and Linux.

- Programmable in C.
- Available accessories.
- Receives IR codes in RC5 format.
- USB interface for easy programming and communication.
- Module I²C bus expansion system.
- Expansion boards may be stacked as needed.
- Sample C programs and huge C function library.
- Powerful tank drive train can drive up steep ramps and over obstacles.
- Large payload capacity.
- Light, collision, speed, and IR-obstacle sensors integrated.
- Two 7.2V DC motors.
- 625 CPR encoder resolution for precise speed regulation.
- Six PCB expansion areas.
- One year warranty.

For further information, please contact:

Global Specialties

Website:
www.global specialties.com

If you have a new product that you would like us to run in our *New Products* section, please email a short description (300-500 words) and a photo of your product to:

newproducts@servomagazine.com

Recycling & Remarketing High Technology
WEIRDSTUFF® WAREHOUSE
Software,Computers,Electronics,Equipment,Doohickies
384 W. Caribbean Dr.
Sunnyvale, CA 94089
Mon-Sat: 9:30-6:00 Sun: 11:00-5:00
(408)743-5650 Store x324
WE BUY AND SELL EXCESS & OBSOLETE INVENTORIES!
FREE COMPUTER RECYCLING
We recycle computers, monitors, and electronic equipment. M-Sat 9:30-4:00
GREAT DEALS!
Hi-tech items, electronics test equipment, and more!
GIANT AS-IS SECTION
10,000 sq. ft. of computers, electronics, software, doo-hickies, cables, and more!
also check out our...
eBay Store
stores.ebay.com/WeirdStuff-Inc
WWW.WEIRDSTUFF.COM

ALL ELECTRONICS CORPORATION
Thousands of Electronic Parts Available Today!
LEDS · CONNECTORS · RELAYS
SOLENOIDS · FANS · ENCLOSURES
MOTORS · WHEELS · MAGNETS
PC BOARDS · POWER SUPPLIES
SWITCHES · LIGHTS · BATTERIES
and many more items...
We have what you need for your next project.
www.allelectronics.com
Discount Prices · Fast Shipping

SERVO Covering the world of personal robotics
SERVO MAGAZINE
Online Store - www.servomagazine.com
SERVO'S Online bookstore has more than 30 titles on robotics!
Robot Builder's Bonanza
123 ROBOTICS EXPERIMENTS FOR THE EVIL GENIUS
MECHANISMS MECHANICAL DEVICES SOURCEBOOK
Making Things Move
Just to name a few!
(800) 783-4624

SDP/SI
STOCK DRIVE PRODUCTS/STERLING INSTRUMENT
www.sdp-si.com
FREE CATALOG
800.819.8900
GEARS
BELT & CHAIN DRIVES
PULLEYS
COUPLINGS
BEARINGS
SPROCKETS
SMALL MECHANICAL COMPONENTS

"KILL-A-WATT METER"
With summer around the corner and electricity usage going up, don't get caught spending money on appliances that you believe are OFF!
This meter is the best way to check for the "phantom draw" in OFF appliances and to determine your actual energy draw in ON home appliances.
Only $29.95
To order call
1 800 783-4624 or online www.nutsvolts.com

EVENTS Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: **http://robots.net/rcfaq.html**.

— R. Steven Rainwater

JUNE

1 **Concurso Robótica Radio Servicio Aguilar**
Tapachula, Chiapas, Mexico
See website for details and rules for this contest.
http://electronica-aguilar.blogspot.com

1 **Pobot Junior Cup**
Sophia-Antipolis, France
A competition for waste collection robots.
www.pobot.org

4-6 **NASA RASCAL Robo-Ops**
Johnson Space Center, Houston, TX
Contestants build teleoperated planetary rovers. The rover must compete in Houston, TX while being teleoperated by their teams from their home university locations.
www.nianet.org/RASCAL

4-9 **NASA Sample Return Robot Challenge**
Worcester Polytechnic Institute, Worcester, MA
Robots must navigate unknown terrain, avoid obstacles, and return a sample.
http://challenge.wpi.edu

7-9 **National Underwater Robotics Challenge**
Chandler, AZ
Student ROV contest.
www.h2orobots.org

7-10 **AUVS International Ground Robotics Competition**
Oakland University, Rochester, MI
Autonomous ground robots navigate an outdoor obstacle course within a prescribed time limit.
www.igvc.org

8 **Robotic Day**
Prague, Czech Republic
Some unusual sounding events like Bear Rescue and Ketchup House, as well as traditional robot events like line following and mini Sumo.
www.roboticday.org

8 **Sparkfun Autonomous Robotics Challenge**
Boulder Reservoir, Boulder, CO
Autonomous ground and air robot contest.
http://avc.sparkfun.com

20-22 **MATE ROV Competition**
Weyerhaeuser King County Aquatic Center Federal Way, WA
Underwater robots navigate a course with a goal that varies each year. See the website for details on this year's challenge.
www.marinetech.org

26 **HBRC Challenge**
Mountain View, CA
TABLEbot competition. See the HBRC website for all details.
www.hbrobotics.org

29 **UK National Micromouse Competition**
Birmingham, United Kingdom
Small autonomous robot mice race through a maze to win the coveted brass cheese award.
www.bcu.ac.uk/tee/events/techfest/micromouse

29-30 **International Autonomous Robot Contest**
Reuben H. Fleet Science Center San Diego, CA
Autonomous robots must navigate a course around fixed obstacles.
www.iaroc.org



hardware made easy

Solutions Cubed

www.solutions-cubed.com designservices@solutions-cubed.com 530.891.8045

for a 5% discount use coupon code SERVOMAG13 (through 1/1/14)





SUPER SIMPLE SWARM SLINKYS

Swarm Slinky is a project at the Technical University of Madrid. The idea behind Swarm Slinky is to build a multi-robot system consisting of six independent units designed to explore difficult terrain.

The six units are interrelated, creating a mobile network distributed by robots which are able to transport payloads (sensors). The swarm can be spread out from a central point (supply rover), then move in a radial sequence with the purpose of moving as far away from the rover as possible. The obtained effective range from the point of view of the exploration is the combination of the six ranges of each individual robot.

In addition to this capacity to cover an explored length bigger than the one which would be covered by a single robot, it should be noted that the six robots will form an irregular hexagon which defines a zone where the measurement of determined variables can be done.

Finally, the advantage of using a swarm of robots must be emphasized, such as the increasement in strength of the system; a possible breakdown of any of the individual robots would not affect the mission totally.

The given system can be divided in two levels: collective and individual. On the one hand (as a group), the individuals are considered as identical and simple units which can be dispensable for the development of the mission. Their capacities of intercommunication allow them to act as nodes of an interrelated network; a mesh. If some of the units get damaged, the system — as a group — can continue to operate.

The four legs of each robot have independent movements. They are connected to the body by four continuous rotation drives with position feedback. This configuration is very versatile and allows different locomotion modes such as slinky, rolling, caterpillar, quadruped and whegs.

Slinky and rolling have been selected as the principal means of locomotion. The rest of the modes can be used to overcome different obstacles.

Swarm Slinkys are simple, strong, and economic.



COMAN THE ROBOTARIAN



Most humanoid robots developed over the past few decades have had stiff joints. That's a problem if they're ever going to interact with people. Their unyielding arms and legs could cause an injury if they accidentally whack someone, or if they lose balance and fall down. Lately, there's been a growing interest in developing robotic joints with variable stiffness which would improve their safety, but so far, few groups have built a complete robot. Now, however, a team from the Italian Institute of Technology (IIT) is approaching that goal with their robot COMAN (COmpliant huMANoid).

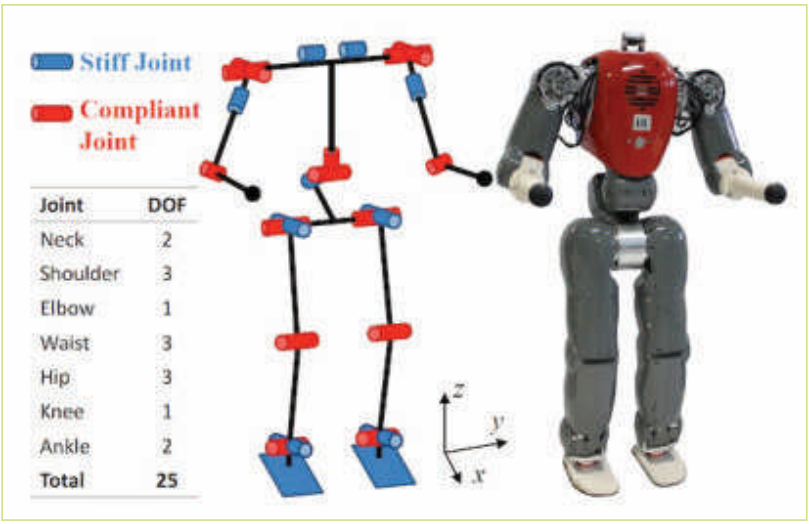
Modeled after a four year old child, COMAN is 94.5 cm tall (from foot to neck) and weighs 31.2 kg. It features 25 degrees of freedom and a combination of stiff and compliant joints. The compliant joints (14 DOF) rely on a series of elastic actuators. These actuators — a custom design created by the IIT team — are applied to the flexion/extension of the arms and legs, and are both small and modular which makes them ideal for multi-DOF robots like humanoids. The researchers have also developed custom torque sensors for each of the elastic joints, including a six-axis force/torque sensor for the ankle joints.

So, what exactly does this compliance get you? The elastic actuators literally add a spring to COMAN's step. In walking experiments, the robot's hardware naturally absorbed the ground reaction forces of each footstep "without additional control enforcement, which is difficult to be realized by the stiff actuated humanoids if no particular foot mechanism or active control is applied." When the team implemented a stabilization control method, the robot steadied itself on a moving platform and when it was knocked around.

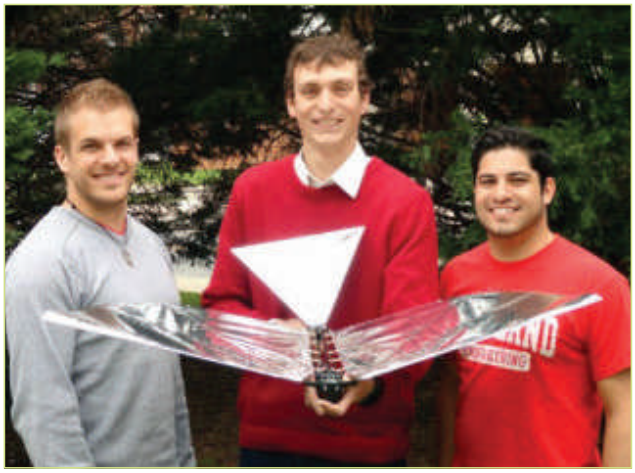
COMAN's internal structure is made of titanium alloy, stainless steel, and aluminum alloy, covered by an ABS plastic exoskeleton.

In the course of developing their own compliant humanoid, the team also devised a method for determining the optimal joint elasticity which, until now, has largely been a time-consuming trial and error process with scant documentation. Their method provides a framework for other researchers exploring compliant robots, and is based on resonance analysis and energy storage maximization criteria.

The IIT team — which includes Nikos G. Tsagarakis, Stephen Morfey, Gustavo Medrano Cerda, Zhibin Li, and Darwin G. Caldwell — is among the first to build a compliant humanoid with both arms and legs. If you're wondering why the robot is headless, don't worry. A head is in the works. Plus, researchers have also completed a pair of hands.



Joint	DOF
Neck	2
Shoulder	3
Elbow	1
Waist	3
Hip	3
Knee	1
Ankle	2
Total	25



ROBO RAVEN TAKES FLIGHT

Leonardo da Vinci would have been fascinated by the work of scientists from the University of Maryland who recently took a major step towards artificial bird-like flight.

Professors S.K. Gupta, Ph.D. and Hugh Bruck, Ph.D., have built the first-ever bird robot with independently-moving wings. This development makes the robot's flight so acrobatic and maneuverable that their "Robo Raven" can do dives and flips, and was even attacked by a hawk. To make their robot feasible, Gupta and Bruck not only had to engineer all of the robotic control systems to behave like a bird, but they also had to use advanced manufacturing technology such as 3D printing with design inspirations drawn from nature.



(NOT SO) TINY DANCERS

Australian artist and builder Geoffrey Drake-Brockman wants to create four life-size robot ballerinas and teach them to dance the "Coppelia" ballet which is about a clockwork girl. Why make robot ballerinas? Well, lots of reasons. According to Brockman:

• It's spooky and beautiful. These robots look a bit like scaled-up clockwork ballerina music boxes with an added "cyborg" factor. They are very strange, eerily-attractive devices that fall outside any regular category.

• It's awesome technology. Here are handmade robots developed from scratch without any assistance from NASA or Honda. They use all custom circuit boards, firmware, communications busses, control software, lasercut aluminium skeletons and motorized joints, and a unique and beautiful form based on a body mould of a real live prima ballerina.

• It's art that's just gotta be done. It's an extension of a long-standing series of art projects with robotics that I have been running for several years, starting with "Floribots." Humanoid robots are the ultimate pinnacle of robot making, and here we can make a batch of them and see what they look like dancing!

• It's important. I think that by dealing with robots at this level – artistically – we can better work out how we feel about them, what their limitations are, how robots fit into the human world. Other robot projects are military oriented or only deal with the "geek" aspect. Here, we are looking at the graceful, beautiful, other possibility of robotics.

The Coppelia Project robots are specially designed to learn and perform the movements of classical ballet. They can spin "en pointe," move their waists, arms, and head. They cannot walk, and their hands do not have grippers to pick things up. They are optimized only as ballerina robots.

The Coppelia robots are taught ballet movements by having their arms, head, and torso physically moved through a ballet sequence by a ballerina trainer. An onboard computer captures the motion so it can replay it later, in various dance move combinations.

Brockman obtained the support of the WA Ballet, a major national ballet company. With the assistance of its principal dancer, Jayne Smeulders, a full set of body moulds of a real ballerina standing en pointe were cast. Much research has been done with Smeulders into the movement requirements for robotic ballet.

A working set of prototype robotic arms has been handmade and tested with the electronics and software to achieve human motion capture and replay.

Two ballerina robots are already partly assembled (at the time of print). One is complete with high-gloss lacquer paint and a fully polished aluminium skeleton; the other one has unpainted panels and an unpolished skeleton. There are silicone molds to cast the rest of the body units.

CAD design files are utilized so more aluminium skeleton parts can get laser-cut. Circuit boards and embedded software have been developed and tested. Sufficient boards for four robots have been assembled and Flashed with software. A fully customized motion editor and joint control suite of software

ASK NAO

Nao is a busy robot. The little humanoid plays soccer, helps with research, works at hospitals, performs comedy routines, and even grooms cats. Now it's going to school, to help children with autism and special needs.

Aldebaran Robotics recently announced the launch of ASK Nao (Autism Solution for Kids) — a program that will pair its little humanoid robot Nao with children with autism. The company hopes to build on the success of previous studies which showed that some children with autism achieved a 30 percent increase in social interactions and better verbal communication when a robot was in the same room. The studies also showed that these improvements can extend to interactions with parents and therapists.

"We have developed educational games that allow children to work on verbal and non-verbal communication, emotional intelligence, mimicking, and even basic academic



skills," explains Olivier Joubert, autism business unit manager at Aldebaran. "After a year of testing, gaining the positive feedback from our beta testing schools in Britain and the United States, and the encouragement of the autism community, we have been driven to launch this initiative to help children with autism throughout the world realize their full potential."

has been developed from scratch and tested.

Brockman also has sufficient motors, bearings, mechanical fittings, wiring components, CPUs, and most of the other electrical parts to finish four ballerina robots. Completing the robot ballerina dance troupe will go as follows:

- Lasercutting, hand-machining, and assembly of two additional robot skeletons.
- Production of two additional fiberglass bodies, hand finishing, and lacquering.
- Installation of electronics and wiring into all four robots.
- Setup of software and integration of systems to enable dance capture and replay.
- Training of the robots to perform ballet moves.
- Software development and hardware refinement to create a cohesive robotic dance installation.





- Production of dance sequence videos of the robot ballerina troupe for publication on YouTube.

The Coppelia Project is the ultimate outcome of a series of increasingly ambitious robotics projects for Brockman. In 2005, he created a work called "Floribots" incorporating 128 robotic flowerpots. This artwork has been exhibited internationally and started the process that led to this project. He has also completed other major robots, including the highly complex spatial robot "Headspace" in 2010 (with 256 motorized mirror-polished rods) and "Totem" in 2012 — the world's largest laser projecting robot at 35' tall and weighing almost 20 tons, with 108 moving elements and three laser projectors.

Two other ballerina related projects have also taken place alongside the Coppelia Project. One is "Parallax Dancer" – a 3D virtual dance installation; the other is "Cockwork Jayne" – a simple windup version of the ballerina robot.



BIONIC BUGS

The last time dragonflies of this size roamed the earth, dinosaurs were still about 100 million years down the line. Insects can't get that big anymore because there simply isn't enough oxygen in the air to keep them going. That's why we have robots: to resurrect freakishly large bugs and make them do our bidding.

Festo — a company that invents amazing bio-inspired robots out of nowhere — have just announced BionicOpter: a scarily impressive robotic dragonfly.

With a wingspan of 70 cm and a body length of 48 cm, the model dragonfly weighs just 175 grams. The wings consist of a carbonfiber frame and a thin foil covering. The structure is made of flexible polyamide and terpolymer. The small ribcage houses the battery, nine servo motors, and a high-performance ARM microcontroller — all installed in the smallest of spaces, just like the sensors and wireless modules.

Up and down, forward, backwards, and to the side, the flapping wing design of the BionicOpter enables it to fly in all directions in space and hover in mid-air, just like a helicopter. Unlike a helicopter, though, the dragonfly does not need to tilt forward to generate forward thrust. This means that it can fly horizontally, as well as float like a glider. Its lightweight design means it is able to start autonomously.



WHERE'S HUBO?

Later this year, some of the world's most advanced humanoid robots (and their human masters) will gather for the DARPA Robotics Challenge (DRC) — a competition where the robots will attempt to perform a series of complex tasks in a disaster response scenario. The highly anticipated event is still several months away, but teams will have to show that their robots can perform adequately in a computer simulation in June. Teams are working frantically on their robots and simulations, and while some groups operate in total secrecy, others like Team DRC-Hubo are eager to show off their progress.

The DRC will consist of eight tasks, including some that are highly complex — such as driving a vehicle or climbing a ladder. Any one of these activities would be difficult to prepare for individually — let alone altogether — which means that, if the robots prove successful at even some of the tasks, the DRC will be considered a major milestone for robotics. It will also serve to showcase the practicality of the humanoid form-factor which some have dismissed as expensive research projects unfit for real world use.

Team DRC-Hubo is a consortium of several universities, including Drexel University and KAIST's HuboLab (and its spin-off RAINBOW Co., which markets its robot technology).

Cool tidbits herein provided by www.botjunkie.com, www.plasticpals.com, http://www.robots-dreams.com, and other places.

A NICE GESTURE



Actroid-SIT — a lifelike robot from Japanese firm Kokoro — hasn't received as much attention as her cousin, Geminoid F, which happens to be a copy of a real woman. While Geminoid F is a teleoperated robot, Actroid-SIT can function autonomously — talking and gesturing while interacting with people. In fact, researchers have recently demonstrated how improvements to Actroid's behavior can make it look smarter and more expressive than your average android.

Actroid makes eye contact and gestures in the direction of a person trying to speak to her, allowing it to adeptly handle crowds of people. To develop the new behavior, researchers from Nara Institute of Science and Technology studied how individuals and groups interacted with the robot. Based on their observations, they focused on two new features which they call "interruptibility" and "motion parameterization," hoping to improve human-robot interaction.

The first feature lets Actroid-SIT handle interruptions in a graceful way. In their human-robot interaction experiments, the researchers noted that interruptions occurred about 26 percent of the time. For example, the human switched topics or handed off the speaking role to someone else. The problem is that despite these interruptions, the robot would obliviously carry on until it finished its spiel. Totally not socially elegant!

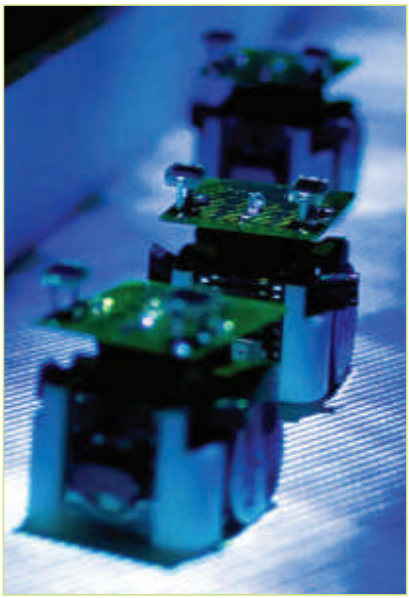
With the new interruptibility feature, the robot can immediately end its current topic and elegantly transition to the new response. Those few seconds count — people interacted significantly longer with the robot when it was interruptible.

Actroid-SIT's motion parameterization system gives her 18 gestures like pointing or waving to let her adapt to the location of the speaker, making the person feel like the robot is really paying attention to him or her.



So, even though talking with the Actroid is still far from a natural conversation, the researchers say this improvement makes a big difference in how people perceive the robot. Participants called the android more friendly, sensitive, sophisticated, and warm when the new gesturing system was used, compared to a normal gesturing approach.

ALICE IN WANDERLAND



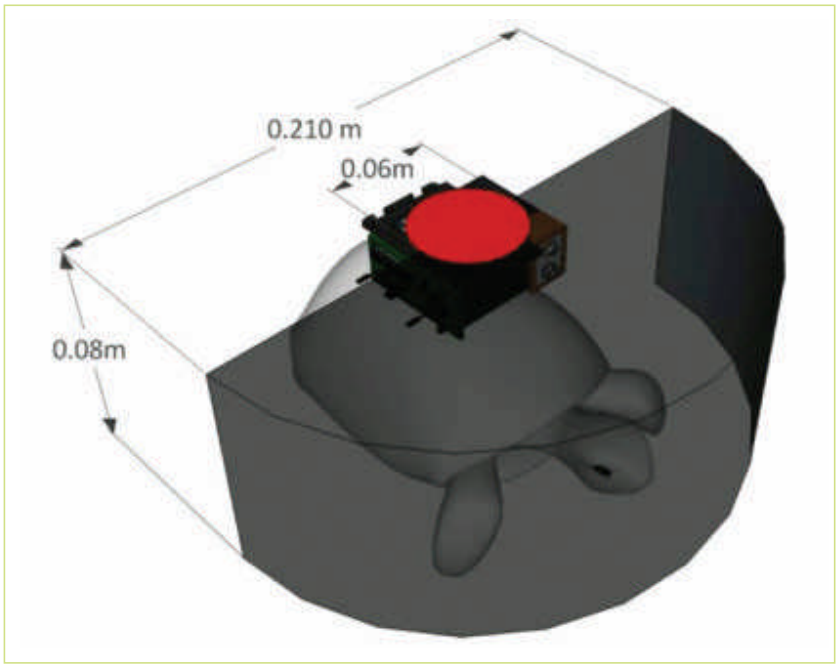
Insects are masters of the swarm. Bugs like bees, termites, and ants manage to do all sorts of complicated and productive things, despite the fact that on an individual level, each insect is really not that smart. So, creating complex behaviors from simple systems is appealing to roboticists. However, researchers from the New Jersey Institute of Technology in Newark, and at the Research Centre on Animal Cognition in Toulouse, France, are using swarms of ant-like robots to efficiently navigate networks without any sort of cleverness at all.

These robots are called Alice — either collectively or as individuals. Their behavior is based on Argentine ants and, as such, they have very, very primitive sensing systems — little more than a couple of light sensors. The little robots were released into a simple network maze where they wandered around a bit looking for an objective while trying to take the least number of turns possible. Wherever they went, they left a trail of "pheromones" as lights turned on above them. All of these behaviors are what ants do, and it turned into a very effective way of autonomously discovering an efficient path.

It appears this research wasn't intended to be specifically about robots, but rather to use robots to try and figure out how ants manage to do ant-y things despite having tiny brains and lousy eyesight.

"This research suggests that efficient navigation and foraging can be achieved with minimal cognitive abilities in ants," commented lead author Simon Garnier. "It also shows that the geometry of transport networks plays a critical role in the flow of information and material in ant, as well as in human societies."

TURTLES times TWO



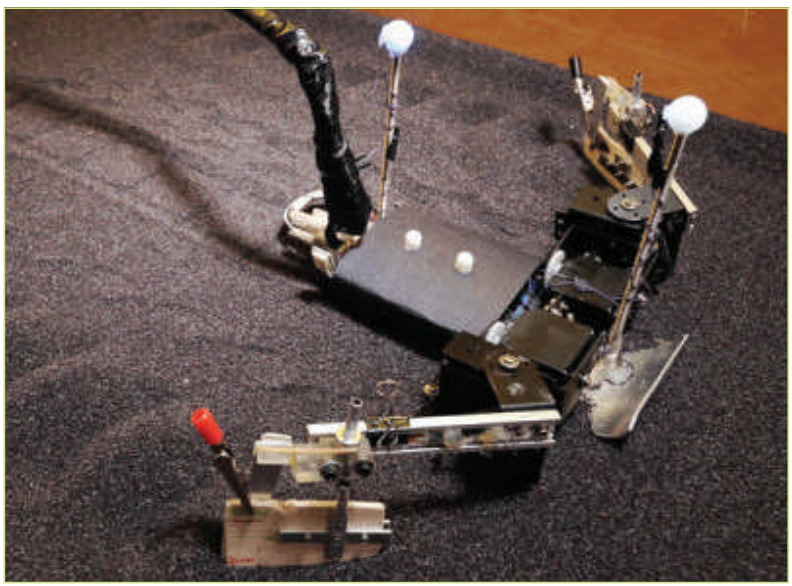
So, we've turned insects into cyborgs using nerve stimulation. That's pretty cool and all, but what's cooler are turtles. Researchers at KAIST in South Korea have managed to hack a live turtle, adding a noninvasive steering system that they've successfully used to get the animal to follow an arbitrary winding path. (Yes, this means we can have cyborg turtles now.)

The concept here is so absurdly simple it's amazing we don't have remote control pets already. The turtles (they're red-eared sliders) get an attachment to their shells consisting of a half-cylinder that can be remotely rotated with a servo. By rotating the half-cylinder around to present the turtle with what looks like an obstacle on one side or another, the turtle can be encouraged to move towards whatever direction appears to be obstacle-free.

The turtle has not been conditioned or trained in any way; the cybernetic control system is just tapping right into the turtle's innate instinctive behavior of obstacle avoidance, allowing voluntary control of the turtle. The word

So, think we need more robotic turtles? Well, you're in luck thanks to this robotic baby sea turtle sorta thing. It's called Flipperbot, and it's designed to help biologists figure out how animals with flippers move in sand, so roboticists can figure out how to get robots with flippers to do the same.

In general, we humans don't have a lot of firsthand experience in using flippers in sand, but that's okay because plenty of other animals do, and sea turtles are among the cutest of those. They've been busy flippering around in sand for millions of years, so rather than attempting to derive optimized flippery movement from scratch, roboticists are "cheating" by simply trying to duplicate what the turtles do, and determine what makes that technique the one that the animals have converged on.



This, of course, requires a robotic sea turtle that is not quite as cute as the real animal, and roboticists from Georgia Institute of Technology and Northwestern University went ahead and built one.

"On soft sand, the animals move their limbs in such a way that they don't create a yielding of the material on which they're walking," explained spokesman Daniel Goldman in a recent commentary. "That means the material doesn't flow around the limbs and they don't slip. The surprising thing to us was that the turtles had comparable performance when they were running on hard ground or soft sand."

The key to maintaining performance seems to be the ability of the hatchlings to control their wrists, allowing them to change how they use their flippers under different sand conditions.

"On hard ground, their wrists locked in place, and

"voluntary" means that the turtle isn't being forced to go in any particular direction; it's just deciding to head where it doesn't see any obstacles.

So, what's so great about turtles? Well, they can move around in land and water, and they're powered entirely by whatever plants happen to be around — which is more than we can say for any robot currently in existence. As the researchers put it: "The system is suitable for application in tasks traditionally carried out by mobile robots, such as surveillance and reconnaissance, exploration and navigation, as well as other missions dangerous for humans."

they pivoted about a fixed arm," Goldman explained. "On soft sand, they put their flippers into the sand and the wrist would bend as they moved forward."

"In the robot, the free wrist does provide some advantage," said Goldman. "For the most part, the wrist confers advantage for moving forward without slipping. The wrist flexibility minimizes material yielding which disturbs less ground. The flexible wrist also allows both the robot and turtles to maintain a high angle of attack for their bodies, which reduces performance-impeding drag from belly friction."

It's interesting to compare the movements of sea turtle hatchlings to the movements of sea turtle adults because — at first glance — the gait that the robot is using (pushing with two flippers at once) seems to be the gait that the much heavier adults favor, while the hatchlings prefer to alternate flippers for speed.

While Flipperbot itself is not going to be heading out to the beach anytime soon, there are already beflippered robotic sea turtles out there that may be able to benefit from this research directly such as ETH's Naro-Tartaruga robot which is designed for swimming and not for crawling around on land. It's entirely possible that sea turtle wrists play a significant role in swimming efficient, too. If Naro-Tartaruga could one day make it onto a beach, it would add a huge amount of versatility to the robot.



COMBAT ZONE

Discuss this section in the *SERVO Magazine* forums at http://forum.servomagazine.com

Featured This Month:

28 *BUILD REPORT: The Quest for a Different Kind of Flipper: Improving the Trigger*
by Zac O'Donnell

31 *Then and Now — A Decade Later With Brian Nave*
by Kevin M. Berry

34 *EVENT REPORT: FIRST Orlando Regionals: A New Set of Heroes Emerges*
by Collin Berry

37 *CARTOON*

38 *Tips from the Pits: Speeding up a Motor Change*
by Pete Smith

www.servomagazine.com/index.php?/magazine/article/june2013_CombatZone

BUILD REPORT:

The Quest for a Different Kind of Flipper: Improving the Trigger

● by Zac O'Donnell

In last month's issue of *SERVO*, I wrote about my first two attempts to build a flywheel-powered flipping robot; the latter of which competed under the name Reclipso.

This time, I will talk about some of the difficulties I faced with those two attempts and how the next design addressed some of them. I will also recount my experiences while testing, and the steps I took to mitigate the issues that came up.

The first and most limiting drawback of Reclipso was its inability to drive while the flywheel was spinning. This was caused by an oversight when I selected the weapon controller.

The controller I purchased only supported automatically detected cutoff voltages for lithium polymer batteries. This caused a problem because I was using the small 1,100 mAh A123 cells which are a lower voltage than lithium polymer batteries.

This caused the weapon controller to go into safety cutoff mode as soon as the battery voltage dropped from the current draw of the drive motors.

Reclipso also could only fire the weapon a single time before the trigger mechanism jammed. While I always intended for the system to be able to disengage and fire several times, a lack of testing time resulted in a one hit wonder.

This behavior was only a slight improvement over the original prototype trigger mechanism which could not even engage the weapon reliably a single time.

Reclipso used a steel hook to grab the flywheel and pull the arm through its stroke, but the problem was that the steel hook did not *unhook* from the flywheel once the arm reset.

Now that I had two failed attempts under my belt, I decided to soften my lofty goals of

making a powerful flipper and just try to create a flipping mechanism that could fire three times without any human intervention.

Another reason that Reclipso was unable to really toss an opponent was the very limited gear ratio between the mass in the flywheel and the opponent. The hook and strap implementation of the trigger meant that the rear bar of the flywheel attempted to rotate at the same speed as the flywheel itself.

This meant that the flywheel's momentum was being transferred to an opponent that weighed ten times as much without any reduction in between. In fact, because the flipper was a four bar linkage, the effective gear ratio was worse than 1:1.

The final result was an inefficient momentum transfer and very little actual flipping.

Now that I had identified the problems with Reclipso, it was time to solve them. The weapon speed controller was simple — all I had to do was find one with a user programmable low voltage cutoff and fit it into the robot. There were many to choose from, and I eventually settled on one that could handle up to 40 amps.

Unfortunately, the speed controller cutoff was the only problem that could be solved simply by replacing a single component. The other problems all required a complete redesign of the weapon assembly.

The new design (**Figure 1**) eliminated the steel hook and strap in favor of a friction clutch. To engage the weapon, I used a servo to slide the clutch downward until it lodged between the flywheel and the coil drum. This was done while the flywheel was spinning at full speed which caused the coil drum to spin rapidly, as well. I used Kevlar thread to link the coil drum and the coil plates that drive the rear bar of the flipper.

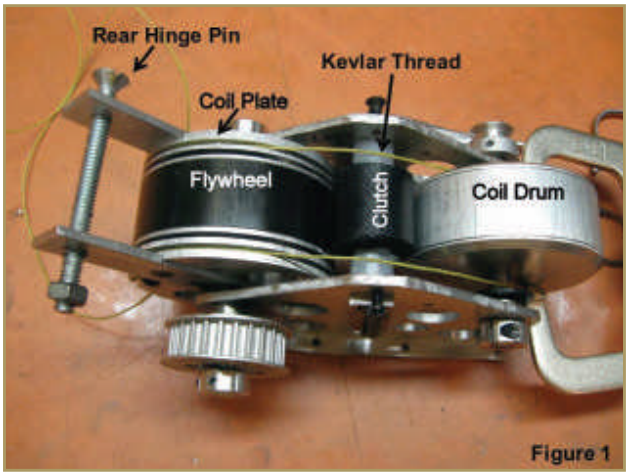


FIGURE 1. Completed assembly with thread.

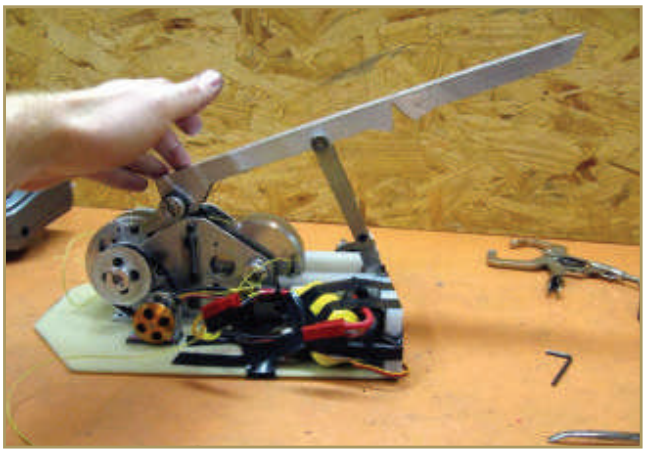


FIGURE 2. Installed extended weapon.

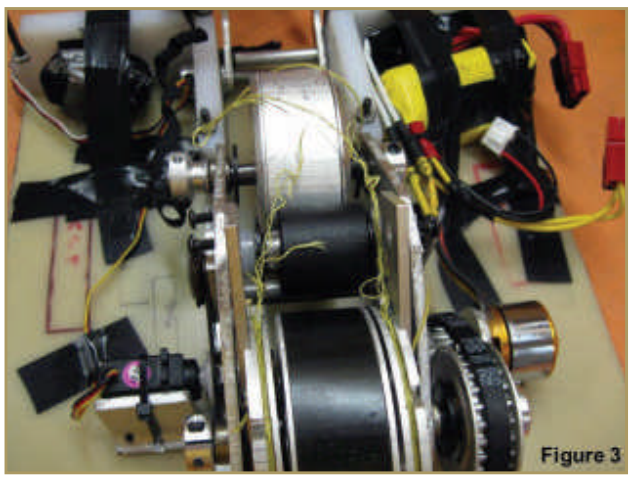


FIGURE 3. Broken Kevlar.

Every rotation of the coil drum reeled in more of the thread and pulled the arm through its stroke. The coil plates at the back of the robot were eight times larger than the coil drum shaft, providing an 8:1 gear reduction to make more efficient use of the momentum in the flywheel.

Once the flipper arm reached the end of its stroke, the thread went tight and everything stopped rotating. The clutch wheel was then pulled back and upward by a pair of springs, so it disengaged from the flywheel and coil drum.

Once the clutch was disengaged, the entire assembly could rotate back to its original position so it was ready to fire again.

There were a lot more moving parts involved in making this design work than there were in the simple steel hook design, but the potential for improved flipping was definitely there.

In **Figure 2**, you can see what the entire assembly looked like with the weapon extended. The flipper arm had not yet been bent to its final form for this picture, but you can see the four bar linkage including the rear hinge pin from **Figure 1** (right next to my hand).

I added rubber stoppers to the rear hinge pin to soften the blow

when the end of the stroke was reached. The large pulley on the left side of the **photo** was driven by the small orange and silver motor below it. This powered the flywheel while the robot jockeyed for position during a fight.

The rear bars of the flipping arm had bearings inside them so the shaft could turn the flywheel while they stayed still.

For this build, I allotted extra time for testing before the competition because I ran into so many problems on the previous version.

Figure 3 shows what happened on the first full power test flip. The Kevlar thread failed when the hard stop was hit at the end of the stroke.

Unfortunately, the Kevlar thread I used was the largest my supplier offered. I needed to find a suitable replacement or a new supplier in time to order the new thread and install it into the robot.

Mike Jefferies — a frequent contributor in the *SERVO* Combat Zone — suggested that I try out a kind of rope called Dyneema. This rope was available in many sizes, ranging from smaller than the Kevlar thread I was using to large enough to tow a car out of the mud. I decided to order a few different sizes and rebuild the robot to fit.

The Kevlar thread I was using was only .040" in diameter. I ordered .075", .095", and .125" thick Dyneema ropes from a spearfishing supplier and set about trying to fit them into the design while I waited for them to arrive.

The trouble with using the bigger rope was that the width between the rear flipper arms could not grow because I did not have time to move the main weapon rails. That meant I had to make space for them by making the flywheel and coil drum narrower.



FIGURE 4. New coil plate.

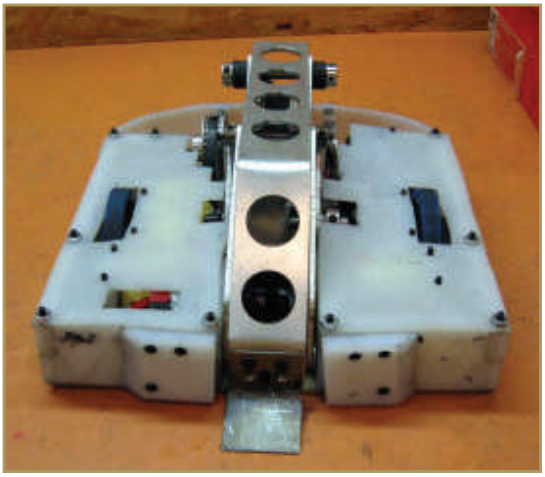


FIGURE 5. Ready for competition.

Once I did that, I had to replace the rear coil plates with wider pulleys that included a place to attach the new Dyneema rope. You can see one of the new coil plates in **Figure 4**.

The new coil plates sported a much wider and deeper channel for rope up to .125" in diameter, and an integrated pin for locking the rope to the coil plate instead of having it stick out the side like I did with the Kevlar thread.

Once I installed all the new parts, it was time for another test. I decided to start with the thickest rope to see if any of the sizes I bought could survive the forces in the flipper.

This time, the new rope held up great, but it took up so much space on the coil drum shaft that it jammed against the insides of the weapon rails. The clutch engaged and disengaged properly but the system did not reset because of the rope jam.

Once I realized what was happening, I decided to switch down to the thinnest rope that I had purchased to see if that would alleviate the pressure. One more test showed that it was strong enough to handle the load while leaving plenty of extra space on the coil drum shaft.

This final version of the weapon assembly proved to be reliable enough to meet my goal of three unattended flips. In the end, the weight from the wider coil plates and the necessity of making the flywheel narrower forced the flywheel mass to be lower than the one in Reclipso.

While this meant that the flipper did not throw other robots as well as I wanted, the clutch mechanism performed flawlessly and that was a huge step in my quest.

I decided to call the new robot Threecoil, and its first competition was the 2012 Bot Blast in Bloomsburg, PA.

You can see the completed machine in **Figure 5**. The robot performed well beyond my expectations at the competition.

The clutch mechanism never jammed and the weapon fired multiple times in each match. I would have been completely satisfied if those were the only results but the robot also managed to win the award for Coolest Robot *and* the first place trophy.

These results were so exciting that I decided to see if I could scale the design up to a 30 pound sportsman class machine for Motorama 2013. You will have to wait for another issue to see how that went. **SV**

Then and Now — A Decade Later With Brian Nave

● by Kevin M. Berry

Last month, the Combat Zone started a new series about the "big names" of our sport during the glory years of television coverage. Now, 10 years after the demise of that major coverage, we tracked down some of the most famous names from that era to gain their perspective after robot combat underwent a drastic change in direction.

Brian Nave qualifies as a big name in so many ways. He participated in every major televised combat series; is co-host of his own TV show on DIY network; organized a successful series of combat events; and is a part of the inner circle of folks who started the Robot Fighting League.

On a personal note, I was honored to be part of the group of people who supported Brian in his BattleBeach and BattleBeach Lite events. Typical of Brian, when a local school needed a "draw" for a technology event, he instantly accepted my invitation to spend a Saturday in a city 50 miles away, helping interest kids in robotics, and was, of course, very popular with the crowds.



Brian Nave, during the height of the BattleBots era.

SERVO: Brian, back in the day you did it all — Robot Wars, BattleBots, Steel Conflict. How did you get into the sport, and what hooked you so thoroughly you sunk so much time, energy, and money into participating?

Brian Nave: I was watching BattleBots™ with my young boys and they were LOVING it. I looked at the robots fighting then — Nightmare, Big Brother, Mauler — and said "I can do a LOT better than that!" I stayed up night and day thinking about different designs.

I loved the excitement and carnage that Mauler brought to the arena, but hated the fact that it constantly broke itself. I was determined to build a spinner that would survive its own power.Thus, Phrizbee was born. As it turns out, surviving its own power was the least of Phrizbee's problems.

Battery and motor technology hadn't quite gotten up to the requirements of this type of bot just yet, and most of my "battles" were getting enough charge into the batteries to last the whole fight and keeping the motors from burning up, but as they say in the robot fighting biz ... "Damage is just weakness leaving the robot."

I kept learning and while I did, more advanced chemistries with higher energy densities became available for batteries, and more powerful motors became available for the spin. I guess what kept me hooked and excited was the constant "destructive testing" of the machine. The never-ending quest for perfection ... I never attained it, but I learned a lot through my failures.

SV: You were at BattleBots during the big TV events. What was it like being a participant?

BN: The "big show" — BattleBots — was the king of robotic excitement back then. No other show could match that energy. The buzz in the pits was electrifying; the high you got when you heard the MC, Mark Biero, announce your bot before the fight was intoxicating; and nothing could beat the feeling of the win.

Unfortunately, wins were few and far between, and just ONE loss bumped you out of the competition. Luckily, just wandering around the pits was an adventure in itself. You got to rub shoulders with all the stars of the show, the robot builders and their creations — all works of art, science, and engineering. I loved wandering the pits and talking to the builders. Everyone was so friendly, so helpful, and SO SMART!

That would have been enough for an aspiring geek like me, but the fun didn't end there! Mixed in with that eclectic bunch of mad scientists were various celebrities and TV stars who were hosting, co-hosting, announcing, commentating, and COMPETING. People like Will Wright who wrote The Sims and other popular computer games; future MythBusters Jamie Hyneman, Adam Savage, and Grant Imahara; computer genius Michael (Fuzzy)

Mauldin who founded Lycos and wrote the search engine; Hollywood beauties Carmen Electra, Traci Bingham, Donna D'Errico, and Heidi Mark; and the geek icon, Bill Nye, the Science Guy.

SV: Same question on Robot Wars in the UK.

BN: Robot Wars in the UK was equally incredible, but an entirely different experience. Where BattleBots was a completely open competition and anyone could enter, Robot Wars was by invitation only. BattleBots was a giant competition that just happened to be televised. Robot Wars was a TV SHOW, period.

For Battlebots, I paid for everything. I paid to enter the competition. I paid to ship my robots, my team, and myself, and I paid for my hotel, food ... everything. If my robots were good enough and made for an entertaining show, then I would make it to the rounds that were recorded for TV and make money from the royalties that BattleBots paid to the builders. As good as the payments were, very few builders made money. (That is why my team motto was "Win or lose, but be spectacular.")

Robot Wars, on the other hand, was completely the opposite. I was paid to appear, they paid for the robot shipping, they paid to fly my team and me, they furnished room and board, they drove us around, and EVERY fight was recorded for broadcast, and did I mention it was recorded in LONDON, ENGLAND!

The builders were mostly pulled from the BattleBots stables, so I got to make deep, lasting friendships with my fellow builders as we spent many hours waiting. Hurrying up and then wait. Then wait some more. Battlebots had 600 robots, and less than a week to get all those fights in.

If you do the math, you'll see that 600 robots/six days/10 hours per day means 10 fights per hour, never stopping for lunch, and going all day long! Since the fights were three minutes long, that meant the builders and crew had three minutes to load, unload, and get the next fight going. Needless to say, the action was non-stop and the schedule was a whirlwind of craziness.

Robot Wars recorded for about a week also, but with 20 or 30 robots. You can see how this would create a LOT of dead time. The silver lining there was that we had time for sightseeing and lots of good times.

Good times, good pay, and no expenses. Robot Wars was great in its own way, and it was CRAZY popular over there. While every fight was recorded for TV, they also had a live audience that paid to watch. They put on about four two-hour shows a day, and the stands were packed for every one — and this was a major stadium that seated 5,000+ people.



They had a huge building that was full of advertisers and activities that the crowd would wait in until their show time. The kids would play video games and see static displays of the robots, and we would go over to sign autographs and meet the fans — many of which knew us and knew our robots!

One experience that I'll never forget was on my second trip to the UK to record *Robot Wars: Extreme Warriors* when I was going through customs to enter the UK. The stone-faced customs officer asked me my purpose in London and I responded that I was there to record *Robot Wars* ... the guy busts out this crazy-fan smile and nearly screams "Robot Wars! I love that show. Which robot is yours?"

I told him "The Revolutionist." To which he brightened even more and burst out with "The Revolutionist! I LOVE that robot! Well, good luck Roboteeer!" Man, if that doesn't make you feel like a rock star, I don't know what will.

SV: You were recognized as the master of the full body spinner (FB) with The Revolutionist, Phrizbee, and Phrizbee Ultimate dominating the sport. You've been quoted many times talking about how it's an optimum design since "the weapon IS the armor." An FBS is a hideously complicated machine to design, build, balance, and drive. What can you tell us about FBS that you wish you'd known when you started?

BN: Ha, what I learned? Don't build one! So much work, so much failure, people love to hate them, and a well built steel box will beat it.

Seriously though, I have a hard time even thinking about building any other type of fighting robot. For a robot that has a weapon (and I think they all should to be spectacular), the efficiency of weight just can't be beat. What did I learn? Hmm. I saw so many spinners lose a tooth or a hammer, or some other part of their rotating mass, then spin wildly out of control. This type of failure was called "Pulling a Mauler" for that robot's spectacular failure when one of his

hammers broke off and the robot became so unbalanced that it actually spun itself upside down.

I vowed my bots would not do that, so I built many small teeth that were designed to take many small bites out of the opponent, and if one or more were lost (and MANY were!), the robot would still spin under control. The design worked well and my spinners never pulled a Mauler, but was it ever a pain to keep welding those teeth back on throughout a competition!

I think for the time and the budget I had, that welding them on was the best solution back then, but in the future I'd think a little harder on a way to have my little shark's teeth be field replaceable.

One tip I'll share with budding spinner builders is that I designed my shell's hub to be compatible with a standard wheel balancer mount. I would take the whole shell down to my local Tire Kingdom and they would put it on their tire balancer and tell me how much weight to put where to make a perfect balance.

I would calculate the proper size 'tooth' to mount and weld it on in the right spot, et voila! Perfectly balanced shell. You should have seen the crazy looks we got from the customer lounge when we had Phrizbee Ultimate's 30 inch diameter, 200 pound shell with its nasty looking shark's teeth spinning up on that thing!

SV: After the demise of the TV era 10 years ago, you put on the hat of an event organizer with your BattleBeach series in the Daytona Beach, FL area. What made you head that direction, and what did you learn about organizing big events with a bunch of ornery, opinionated builders? *(Editor's note, full disclosure: I was one of these critters!)*

BN: Ha, what did I learn? Don't be an event organizer! Seriously though, when BattleBots started winding down many of us still loved to compete. We had a big meeting in Las Vegas with the big shots in the fighting game and started the RFL — the Robot Fighting League.

We started our own local competitions and a year or so later, Jon Vandervelde, who ran MechWars up in Minnesota; Steve Brown, who ran Steel Conflict in California; and I, who ran BattleBeach in Florida decided we needed a National Championship.



Hammering out Phrizbee Ultimate's shell with the hammer off of Mauler during BattleBots 5.0.

We worked together to create the Triangle Series National Championships (the triangle was made by connecting the dots on a map of where each of our events were held). We invited robots who had placed first, second, or third in any RFL sanctioned event to compete for the ultimate bragging rights. The event was a huge success and we thought we were on our way to big things.

Unfortunately, the economy turned tight and despite consistently high ratings, the networks dropped all the robot shows. Consequently, many people didn't have the disposable income these monstrously expensive creations required, nor the time to fly around the country to compete.

After the TV money dried up, the big bots were no longer viable and the big robot events slowly faded away. Now there's only one left. I won't get into why the TV stations dropped such popular shows back then, but they say time heals all wounds and now — many years later — there are some whisperings of new shows starting, and BattleBot Alumni Mark Setrakian has the new fighting robot show on the SyFy channel.

So, have I learned my lesson yet? Have my fingers been burned enough and my bank account emptied enough? Hmmm. I guess time will tell.

SV: What have you been doing since your fighting and event organizing days ended? Any new ventures you'd like our readers to hear about?

BN: During the robot boon years, I appeared on every robot show on TV: BattleBots, Robot Wars, Robotica, and a few others. I even co-hosted my own show for a while with another BattleBot Alumni Buzz Dawson, called Robot Rivals. I wrote robot articles for *RC Driver Magazine* and I'm now on the editorial board and a contributor to *Robot Magazine*. I dabble in local politics, and coach and referee soccer.

My major source of income has always been industrial robots. I worked for an industrial robotics company for almost 20 years before my partners and I started a new industrial robotic company.

Now I am back to where it all began. I founded a company to work my way through college in 1987,

called LOGICOM Logic Systems (**www.LOGICOMLLC.com**). Back in the 80's and 90's I built computers and networks, and sold components and stuff. These days, LOGICOM is all about cutting-edge industrial controls — automated handling equipment, automated mills, lathes, presses.

We do all phases of industrial automation from the concept to the mechanical design and layout, to the electrical and electronic design, to component specification, to PLC and industrial PC programming, to cabinet building, to installation, to documentation, to training.

I love the design process with industrial automation as much as I did with combat robots. I love taking a boring, accident-prone, dangerous manual job and making it a safe, efficient automated process.

SV: Any last words you'd like to leave with our readers?

BN: I used to complain how much money I wasted on combat robots. The robots themselves, the travel, the time off work, the event arena, and all, literally hundreds of thousands of dollars, but then I realized it wasn't wasted.

I look back at all the great friends I made, all the great times I had, everything that I have learned, and realize that it was an incredible journey.

I only wish that I had the foresight as I traveled the journey to realize how incredible it was, but there was always a schedule to keep, always a goal to meet, always a chore to complete, and I couldn't see the forest for the trees. I single-mindedly pursued that goal with blinders on ... and I missed most of it!

One memory burned into my mind was when my brother and I rounded the back curtain to the main arena of BattleBots. My robots were doing well in Season 5 and we were nearing the finals. We had been rushing around like crazy getting the robots ready, getting in line, making sure of this, and checking that.

I looked at my brother and said "Let's just take a minute to soak this in; we never know when we'll be back."

We stood there and looked at all the lights, all the stars, the TV cameras, the noise, the crowd ... we felt the excitement and drank it all in. Those years of my life were literally filled with times like that and I only saved one. We've never been back to BattleBots.

If there is anything I hope to share with anyone, it would be to not spend so much of your life preparing, planning, and executing that you don't have one. SV

EVENT REPORT:

FIRST Orlando Regionals: A New Set of Heroes Emerges

● by Collin Berry

Another year of FIRST (For Inspiration and Recognition of Science and Technology) Robotics competitions have completed, and I attended an event in Orlando which was co-hosted by the University of Central Florida and NASA. The FIRST competition teaches high school students the value of STEM (Science, Technology, Engineering, and Mathematics, for the three of you who might be confused by that acronym) education. It also aims to get the students working as a team, cooperating, and learning the skills it takes to enter the workforce.

Each year, teams are given a robotics challenge that takes place in the form of a game. Teams have six weeks to design, test, and build their robots in order to compete in this game. This year, the game was called Ultimate Ascent. The game takes place in two stages; two minutes total.

First, the robots (groups of three per side) launch Frisbees at goals that vary in point value. The first 15 seconds of the game is completely autonomous; points are doubled in this period. The second aspect of the game happens at the end. With the clock winding down, the robots attempt to climb a pyramid. There are three levels, again varying in point value. The robots hang from a rung of the pyramid; they must be completely off the ground in order to collect the points.

Teams have various strategies for completing this goal. Some of the speedier robots unload their clips quickly, and then make a mad dash towards the human load zone (a team member is allowed to reload the robot with Frisbees), while others collect loose Frisbees from the arena floor.

Some teams have Frisbee shooters that are able to be aimed; others have static launchers, choosing to focus purely on the highest point goals on top of the pyramid.

The team from Westminster Academy (Shark Attack) used a static launcher, but they used an aiming

FREE TO SUBSCRIBERS
MOBILE APP NOW AVAILABLE!
SERVO
Download NOW On Your Favorite Mobile Device!
FOR ROBOT BUILDERS
iOS • ANDROID • KINDLE FIRE
iTunes
GOOGLE PLAY
AMAZON
OPTIMIZED FOR MOBILE
TABLETS SMARTPHONES

system that involved taking a picture of the pyramid goal; then when a camera sees that image, the team knows they are on target.

For the pyramids, some teams — like the team from Colonial High School (Go Bananas) in Orlando — decided to abandon the higher two levels by using hooks to snatch the lowest level. Other schools, though, were able to climb up the entire pyramid, which was the most exciting part of the game.

The crowd roared to life whenever a robot was

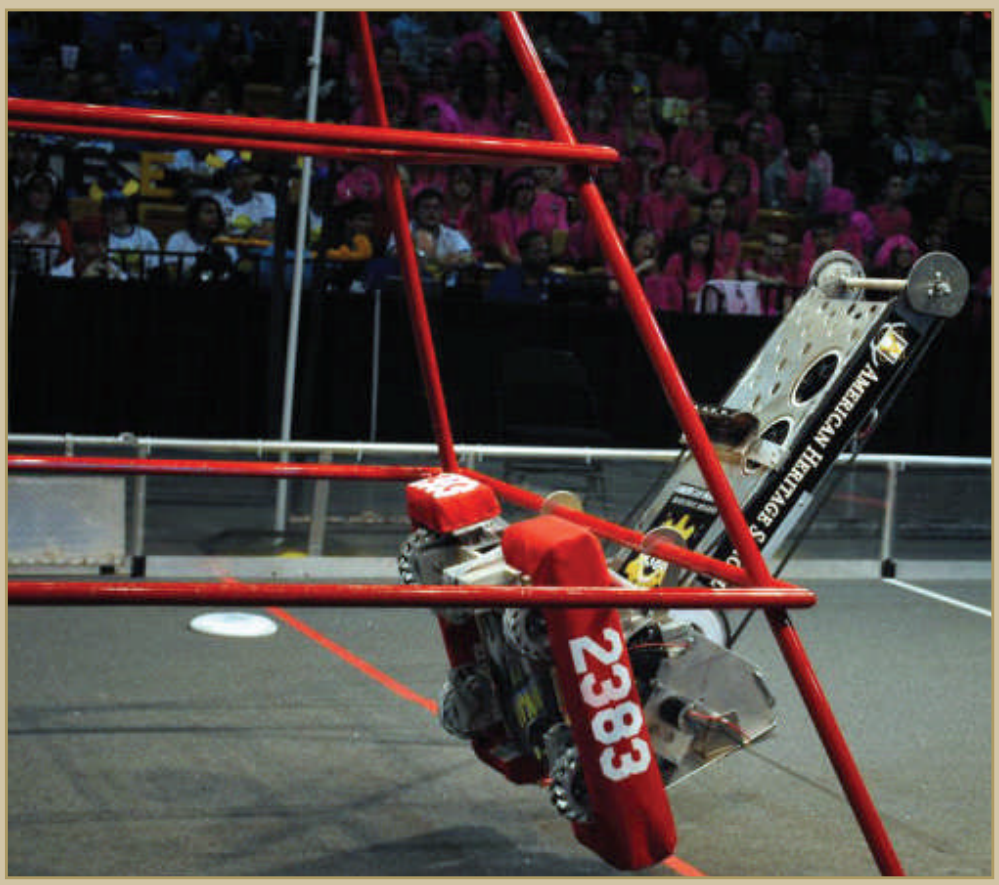


FIGURE 1. American Heritage High School's Team Ninjineers robot hangs from the bottom rung of the pyramid.



FIGURE 2. Team Resistance, Bionic Tigers, and TEAMFORCE are teamed up.



FIGURE 3. The crowd watches the events.



FIGURE 4. A referee checks to see if The Moose's robot is off the ground.

hanging from the top level.

The craziness of the game aside (and it is one of the most entertaining robot games I've ever seen), the purpose of the competition is to educate kids. Speaking to them is actually very impressive. They know the ins and outs of their robots completely. They can name the PSI of their pneumatics; the feet per second their Andy Marks' provided motors give their robot; and they are able to adapt to the challenges of the game. If these kids are the future, the future is in good hands.

The event, however, was eclipsed by tragedy. The night before the competition started, two competing teams (Bionic Tigers and Horsepower) from the neighboring towns of Merritt Island and Cocoa, FL were sharing a trailer to transport their robots to the UCF arena. While stopped at a red light, another driver rear-ended the trailer. One robot was destroyed in the crash, while the other was destroyed when a heavy toolbox fell on it.

Facing an uphill battle, the teams regrouped. Never losing their optimism, both teams returned to their shops (both cities are only a half hour or so from Orlando) to begin the difficult repair session.

They started with parts that could be salvaged from their robots, and then combined those with any spares they could scrap together. Luckily, the teams had mock-ups and CAD designs, so reconstructing the frame was relatively easy. (I mean, how easy is it to complete two months of work in a number of hours?)

Other teams were eager to pitch in any spare parts they could. Team Shark Attack, for example, retreaded the tires.

All the robots are sealed prior to the competition. That means the teams could remake the frames and do the welding required, but could not actually construct their robots before the day the event started — giving the teams just six hours to redo six months of work.

Miraculously, they finished. Both robots were able to not only function, but compete. The teams found themselves in the same alliance — a winning one. Not only did they

FIGURE 5. An aerial view of the pits.



build robots in an extremely brief period of time, but they advanced to the quarter finals before being knocked from the tournament.

These two teams exemplify the spirit of FIRST robotics: teamwork, creativity, and ingenuity. They came together as a team in order to do something that was extremely challenging, then triumphed in adversity.

Winners were: The Brazilian Trail Blazers (Gravatai, Brazil); Team Krunch (Tarpon Springs, FL); and Shark Attack (Ft. Lauderdale, FL). SV



FIGURE 6. The remains of the two destroyed robots.



FIGURE 7. Team Horsepower's robot, before and after.



Tips from the Pits:

Speeding Up a Motor Change

● by Pete Smith

Half the battle in combat robotics is having your bot repaired completely in time for the next fight. This is the first of a series of short articles detailing some ways to make that just a little bit easier.



FIGURE 1. My Dave's Hub.

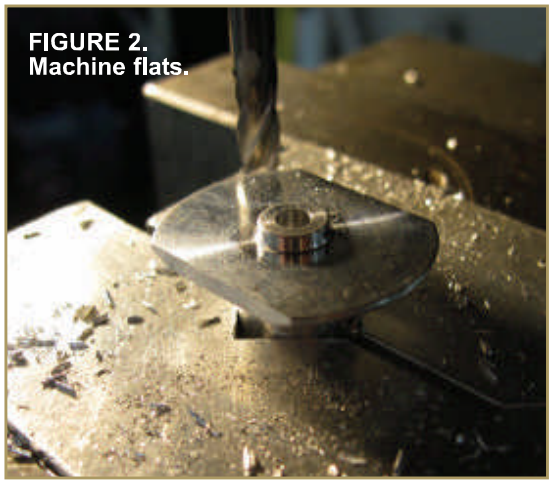


FIGURE 2. Machine flats.

Dave's Hubs (**Figure 1**) are one of the most popular ways to mount Lite Flite wheels on an Ant or Beetleweight bot.

The hubs are secured to the axle with a set screw, and the tire is held on by a custom washer and a screw. It's normal to loctite these screws to ensure the wheels do not come off in combat. However, sometimes a little too much loctite makes the screw very hard to loosen up.

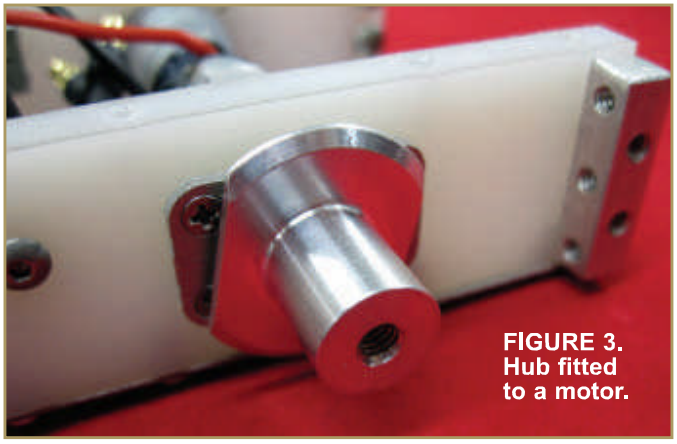


FIGURE 3. Hub fitted to a motor.



FIGURE 4. Using a wrench to hold the hub.

This is made worse by the fact there is little to stop the wheel from simply turning when you want the screw to move. This is the last thing you want to happen if you need to change a tire or a complete motor between fights at an event.

In order to stop a wheel from turning, I machine a flat on either side of the hub (**Figure 2**) so that a slim wrench can be slid down behind the tire and onto the hub. I just happened to have a slim 19 mm wrench, so I machined mine to match that, but 3/4" would probably be an easier size for most in the US. If you don't have access to a milling machine, you could simply use a file to add the flats.

A modified hub can be seen fitted to my Antweight Saifu (**Figure 3**). Notice the other benefit of the modification ... it's now possible to get at the mounting screws for the motor. This is very useful if the set screw on the hub will not come out.

In **Figure 4**, you can see how the wrench fits behind the tire, turning what could be a nightmare in the pits into a quick and easy task. **SV**

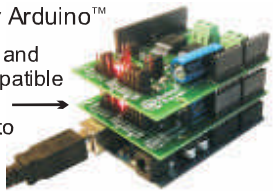
ROBOT POWER™
www.robotpower.com

Extreme Robot Motor Control

High-Current motor control for Arduino™



Software and plug compatible with our MegaMoto shield →



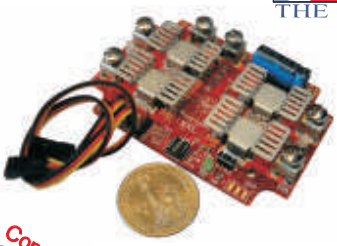
MEGAMOTO *Plus* Shield

- **30A Continuous** with fan - **40A peak**
- Current sensor outputs can be sent to any analog pin
- **Stack** up to three units on one Uno or compatible
- Independent half-bridges can control brushless/steppers too!

MADE IN THE USA

Scorpion XXL

- 7V - 28V
- Dual 40A+ Peak H-Bridges
- Current/Temp limiting
- R/C inputs w/mixing
- Optional enclosure
- 3.25" x 2.25" x .5"





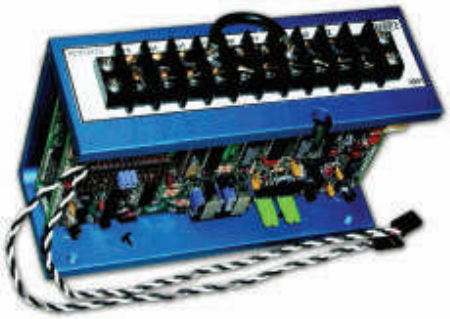
Coming Soon!

The Vyper

- **Single 120A+ H-Bridge**
- 8V - 42V
- R/C inputs w/mixing (2 units req.)
- Analog pot input
- Supports 8 AWG wires!
- Optional enclosure
- 2.85" x 2.25" x 1.75"

Phone: 253-843-2504 • sales@robotpower.com

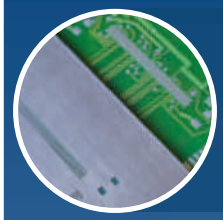
STEER WINNING ROBOTS WITHOUT SERVOS!



Perform proportional speed, direction, and steering with only two Radio/Control channels for vehicles using two separate brush-type electric motors mounted right and left with our **mixing RDFR dual speed control**. Used in many successful competitive robots. Single joystick operation: up goes straight ahead, down is reverse. Pure right or left twirls vehicle as motors turn opposite directions. In between stick positions completely proportional. Plugs in like a servo to your Futaba, JR, Hitec, or similar radio. Compatible with gyro steering stabilization. Various volt and amp sizes available. The RDFR47E 55V 75A per motor unit pictured above.
www.vantec.com

VANTEC Order at (800) 882-6832

THE ORIGINAL SINCE 1994
PCB-POOL®
Beta LAYOUT



FREE Stencil
with every prototype order



NEW!

Embedded RFID
authenticate, track & protect your product
www.magic-pcb.com

Call Tyler: 1 707 447 7744
sales@pcb-pool.com

PCB-POOL® is a registered trademark of Beta LAYOUT create:electronics
www.pcb-pool.com

AndyMark
Inspiring Mobility
Specializing in Unique Wheels, Gearboxes, Aluminum Sprockets and Drive Bases



8" Plastic Omni Wheel

c-Rio-Ready Full Chassis Kit





Aluminum Sprockets

6" HD Mecanum Wheel Set



AndyMark, Inc.
sales@andymark.com
Toll Free: 877-868-4770
www.andymark.com

Open Source *Hardware* for Robot Projects

by R. Steven Rainwater

Did you know that there are community-developed designs for robots that are freely licensed, allowing you to download those designs as a starting point for building your own robot?

To understand the idea of free hardware (or open hardware if you prefer), it's helpful to quickly review the history of the free software movement. Most of us are familiar with the concept of free software as passed on to us by its creator, Richard Stallman, who also started the Free Software Foundation. It's the idea that software should be released under a license, known as the General Public License or GPL, that protects what are known as the "four essential freedoms" of end users:

1. The freedom to run the program for any purpose (including commercial purposes).
2. The freedom to study how the program works, and change it so it does your computing as you wish. Access to the source code is a precondition for this.
3. The freedom to redistribute copies so you can help your neighbor (at no cost or for a fee).
4. The freedom to distribute copies of your modified versions to others, so that the whole community benefits from your changes. Access to the source code is a precondition for this.

What this means in practice is that software can be developed, shared, and improved by a community of users. The software slowly but surely improves over time — even if the original individual or company that started it abandons the project. Others can continue the project or "fork" it (create a new project derived from the first).

The idea of free software requires — among other things — that the source code be openly available to anyone. This idea of "open" source code lead some clever marketing types to invent the name "open source" to market the free software idea to business — sans any mention of user freedom — which they feared would scare businesses, whose business models often rely on restricting user freedoms. This attempt to use the term open source to slightly redefine the underlying principles of free software lead to ongoing political struggles between advocates of each camp. However, both advocated similar things for slightly different reasons.

Over time, an uneasy truce was reached and the combined concepts are now commonly known as FLOSS (Free/Libre/Open Source Software). The FLOSS movement revolutionized our ideas about software. Odds are good that you use and rely on FLOSS every day, whether you realize it or not. Computers ranging from the fastest supercomputers down to most smartphones rely on free software as do most of the computers that make up the Internet.

The success of FLOSS resulted in the application of those same concepts of freedom and openness to artistic endeavors in the form of the Creative Commons organization and its two free licenses, known as the attribution (BY) and share-alike (SA) licenses. So far, everything we've talked about is information; ones and

zeros. In the 1990s, experimentation began with what was called Free Hardware. Like free software, the "free" in free hardware refers to the end user's freedom — not to the price of the hardware. However, unlike the software realm, the terms free hardware and open hardware are synonymous, and carry no political baggage. So, feel free to use whichever term you prefer.

The free hardware idea was to design a physical object; a printed circuit board, a motor, an airplane, or whatever you can imagine. The schematics, mechanical drawings, or other specifications for the design are then released under a license that protects the user's rights; their right to use the design for any purpose, to modify it, and to redistribute it.

One of the early pioneers of this effort was Diehl Martin, known as Marty to his friends. Marty begin designing printed circuit boards that were popular with electronics and robotics hobbyists. He named each new board after a type of breakfast food, such as Donut, Flapjack, or Grits. The designs were released on his website (known as **FreeIO.org**) and users could download the plans and build the boards, or even modify and improve the designs. Software developers collaborated with him to introduce drivers for his boards into the Linux kernel. Within a few years, the idea caught on and everyone was doing it.

Some free hardware designs have now became quite popular — such as the Arduino, BeagleBoard, and Raspberry Pi microcontrollers — and the RepRap 3D printer. You may have heard of the Open Source Ecology group's GVCS project which is releasing open source designs for the 50 industrial machines they deem necessary to maintain civilization (e.g. sawmills, tractors, induction furnaces, and the like).

This brings us to robots. Robots are hardware, so why not create a robot design that can be released under a free license? This has a lot of important advantages for robotic hobbyists and the research community alike. Robots can be expensive and time-consuming to build. Robot builders often spend years duplicating the work of others instead of making steady improvements. As an example, compare the autonomous mobile robots built by William Grey Walter in the late 1940s to robots built by hobbyists today. Little has changed except the materials.

He used vacuum tubes while we use solid-state chips. However, Walter's robots were complex behavior-based designs that could easily win "best of show" if brought to any modern robot club contest.

Part of the reason we haven't gotten better at building robots is that we keep re-inventing and re-building the same basic designs over and over instead of re-using designs and making incremental improvements like we do with software.

When I got interested in the idea of free hardware robotics, I found there was no single reference for locating such designs. So, I set out to find and document them. Many people may not be aware that such designs have existed since as least 1997.

In this article, I've attempted to list the free hardware robot projects roughly in chronological order by the project's creation date. I also had to create some criteria for what to include. Hopefully, my criteria won't seem too arbitrary:

1) The design must be for a complete mobile robot, not just part of a robot such as a manipulator arm.

2) The design documents (e.g., CAD files, schematics) must be available under a free license (i.e., a license that protects the user's four basic freedoms as with FLOSS software — a license with commercial-use restrictions cannot qualify as open).

3) At least one working robot must have been developed and demonstrated. Projects that are in the planning stages didn't make the list. I wanted designs already proven in the real world.

One note about software licenses. Several projects have used a CC license for their software. While this can in some sense be construed as open source, it is against the recommendations of both the Creative Commons organization, the Free Software Foundation, and the Open Source Initiative. Using a CC license on software presents potential legal problems for the end users because the license does not distinguish between object and source code. This is intentional as the CC licenses were meant to apply to other types of works, not software.

Use of a CC license on software also makes that software incompatible with many valid open source licenses. For hobby users, it's a distinction not worth worrying about, but for commercial users it is a serious legal issue and usually means the software will have to be discarded and replaced with new software licensed under an accepted GPL-compatible license.

If you are the creator or maintainer of a free or open project, remember never to use a CC license for the software elements of your project!

The designs provided here meet the three criteria listed previously. Each of these robot projects are available under a license that will allow you to build them, modify them, sell them, redistribute them, or do just about anything else you want with them — provided, of course, that you pass on the same license terms to the next user.

Sea Perch ROV
Date: 1997
Hardware License: CC BY v3.0
Software License: N/A

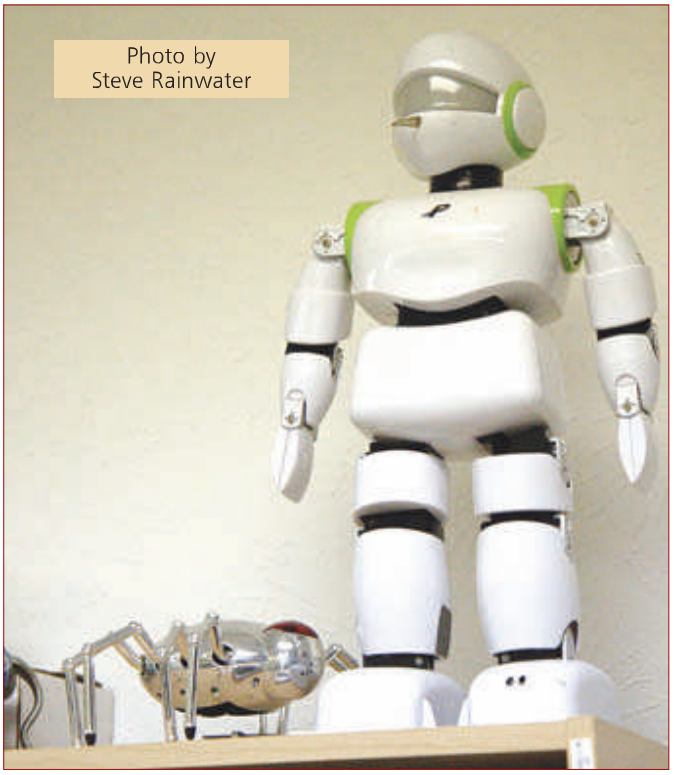

US Navy photo by John F. Williams

The Sea Perch ROV started out as a project description in the spiral-bound book, *Build Your Own Underwater Robot and Other Wet Projects* by Harry Bohm and Vickie Jensen. Later, Professor Thomas Consi of MIT discovered the Sea Perch design and incorporated it into his Ocean Engineering Program curriculum. The MIT project took the original idea and turned it into a simpler and cheaper ROV. MIT also developed parts lists and build manuals for the CC-licensed project. MIT's project was partially funded by the Office of Naval Research, who helped adopt the Sea Perch design as part of a partnership with the Society of Naval Architects and Marine Engineers (SNAME). Through that program, the design was developed into a kit and a contest managed by the Association of Unmanned Vehicle Systems International Foundation. Over 50,000 students have participated so far, perhaps making this the most widespread free hardware design for a robot ever created. For more information, see the ONR Sea Perch website.

The Sea Perch is a remotely operated vehicle (ROV) rather than a true autonomous robot, but the open design can be (and has been) modified to be fully autonomous by the user if desired. Sea Perch kits are available for $143 USD.

Get the design: **http://seaperch.mit.edu/**
Buy a kit: **www.seaperch.org/**

Open PINO Platform Humanoid Robot
Date: 1999
Hardware License: GNU FDL
Software License: GNU GPL


Photo by Steve Rainwater

The Open PINO project was an attempt by Japanese research organizations to establish a freely licensed, standardized humanoid robot platform suitable for research and collaboration. The project started in 1999, with prototypes shown as early as 2000. The robot was 72 cm tall, had 26 degrees of freedom, and weighed 4.6 kg. It could dynamically balance and walk. The control software was based on genetic algorithms. The chassis, boards, and software were released under GPL or FDL licenses. The project description contained this information:

Open PINO Platform is the project to accelerate the research and development of humanoid robots by providing the technical information of PINO open to the public. Everyone can use PINO as a base of the research and development, in other words, to foster PINO to be a more sophisticated humanoid robot. We hope PINO to play the similar role as Linux for humanoid robot development. Please join us to foster PINO!

In 2001, a Japanese company, ZMP, Inc., developed a kit version of the PINO robot and also offered a plastic exterior shell that was not freely licensed. While the robot made headlines for a while, the high cost of the kit combined with the high complexity of reproducing the mechanical and board designs prevented widespread acceptance of the robot. Costs to build a PINO could run as high as $30,000 USD. In 2004, ZMP released a PINO version 2, but the license status is unknown. The version 2 kit was priced at around $45,000 USD.

Get the design: **www.sbi.jp/symbio/symbio/** *(password from ERATO required)*
Buy a complete PINO version 2: **www.zmp.co.jp/e_html/products_pino.html**

iCub
Date: 2004
Hardware License: GNU FDL
Software License: GNU GPL

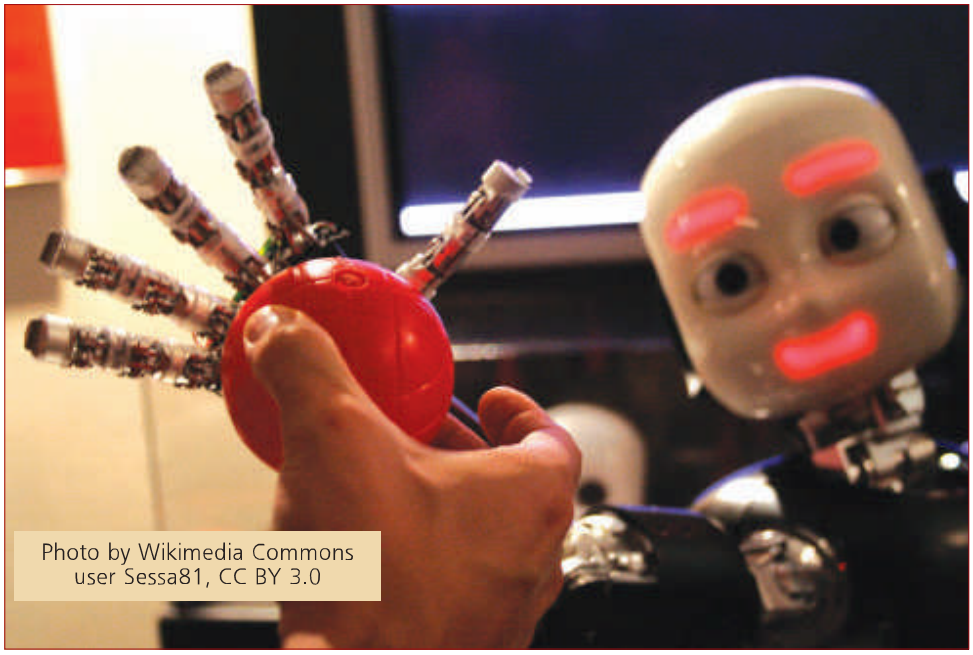


Photo by Wikimedia Commons user Sessa81, CC BY 3.0

The iCub is a one meter tall humanoid robot platform developed by the RobotCub Consortium — a group of EU universities. The goals were identical to the PINO project, but the iCub has met with much more success. Perhaps because of the large number of research institutions that are participating in the project. The robot's hardware, firmware, and high-level software are all under free licenses; in most cases, the GNU FDL and GPL. The RobotCub Wiki is a good source of information on the iCub robot design. As of 2010, over 8.5 million euro has been spent on development of the iCub. The robot has 53 motors — providing twice the degrees of freedom available in the PINO humanoid. Not surprisingly, the iCub is also much more costly to build than the PINO. Assembled units are sold at cost by the Italian Institute of Technology. Cost per unit is about $327,000 USD. Development work has progressed from iCub v1.0 to the current iCub v2.5. An iCub v3.0 is under development.

Get the design: **http://eris.liralab.it/wiki/Main_Page**
Buy a complete iCub: **www.iit.it/en/products/catalog.html**

e-puck Robot
Date: 2004
Hardware License: e-puck Robot Open Hardware License
Software License: e-puck Library License

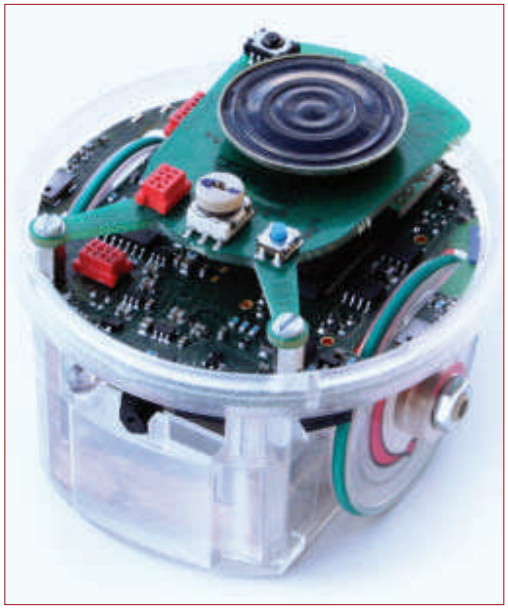


Photo by Stéphane Magnenat, CC BY-SA v3.0

The e-puck project of the Ecole Polytechnique Fédérale de Lausanne is a collaboration between the Autonomous Systems Lab, the Swarm-Intelligent Systems group, and the Laboratory of Intelligent Systems. The goal is to create a free/open miniature mobile robot for educational use that has a simple design, is flexible, user-friendly, and inexpensive. The e-puck is a differential drive robot with a variety of sensors. Many hundreds of e-puck robots have been produced and tested. The design has gone through several releases, with improvements from previous versions. Inexpensive, in this case, means $1,050 USD per robot for assembled units. No kits are available, but all design documents on the project are on the website.

Both the hardware and software are released under one-off licenses. They appear to meet the guidelines of the Free Software Foundation and Open Source Hardware Foundation closely enough to qualify as free/open. However, it's unknown whether they are compatible with standard licenses, so creating derived works that integrate these designs with others under conventional licenses may be problematic.

Get the design: **www.e-puck.org**
Buy a complete e-puck: **www.gctronic.com/e-puck.php**

Jasmine Swarmbot
Date: 2005
Hardware License: GNU GPL
Software License: GNU GPL

The Jasmine Swarmbot project maintains the design for a small (less than 3 cm cube) robot optimized for swarm-related research. Their goal is to maintain a per-robot cost of $133 USD or less. The robots have differential drive, a variety of sensors, and robot-to-robot communication. The robots have been widely used in research projects; in some cases, in swarms of over 100 robots. The design is robust and stable. The Jasmine robot was originally developed at the University of Stuttgart, but the design is now maintained independently. No kits or assembled robots are currently available.

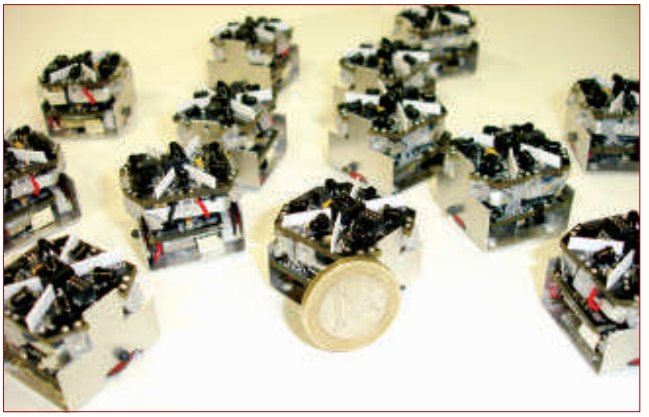


Photo by Jasmine swarm robot development team

Get the design: **www.swarmrobot.org**

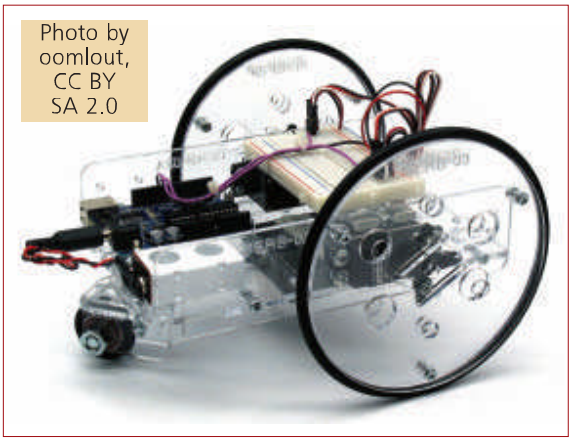


Photo by oomlout, CC BY SA 2.0

SERB
Date: 2008
Hardware License: CC BY-SA v3.0
Software License: CC BY-SA v3.0
(See note in article about CC licensed software)

SERB (Arduino-based SErvo RoBot) was a completely free/open hardware and software robot design created in 2008 by Oomlout, a UK company that developed DIY Arduino-based projects and kits. The robot consists of laser-cut acrylic combined with two RC servos, an open hardware Arduino microcontroller, and assorted off-the-shelf hardware. The resulting differential drive robot uses whisker sensors for impact detection. For a while, kits were available from Oomlout and other online DIY stores. Interest in the design has declined and kits are not available at the time of writing, nor is the design still under active development. Some ideas from SERB were included in a later free hardware design — the Tiny Wanderer robot — also included in this list.

Get the design: **http://oomlout.com/a/products/serb**

Veter
Date: 2010
Hardware License: GNU GPL
Software License: GNU GPL

Photo by Viktor Chernenko, provided by Veter team



A small team of robotics enthusiasts in Germany have been hard at work on a project called Veter for several years. The first public release was made in 2010, and the group has continued to revise and improve the robot. The hardware and software are released under the GNU GPL license and all design files are available from a Git repository. The latest version of the robot is a differential drive tracked robot. It uses a combination of off-the-shelf parts, 3D printed parts, and open source boards like the BeagleBoard-xm. The robot is equipped with ultra-sonic rangers, stereo video, compass, GPS, and other sensors. The robot can communicate via Wi-Fi. It can operate autonomously or can be controlled by a human operator from a navigational interface that includes real time streaming video. The software for autonomous navigation is well developed and utilizes PID control, as well as a Kalman filter and particle filter for position estimation. No assembled units or kits are currently available, but the website suggests that a kit is under consideration and might be available in the future. For more information, see the main Veter website or the Veter blog.

Get the design: **https://github.com/veter-team**

AMIGO
Date: 2011
Hardware License: CERN OHL v1.1
Software License: Various

AMIGO is a service and care robot that is a project of the CST group at Eindhoven University of Technology. The name stands for Autonomous Mate for IntelliGent Operations.

The robot is roughly a humanoid torso mounted on a rolling platform. It stands about 135 cm tall and weighs around 80 kg. It uses three PC type motherboards and other off-the-shelf hardware, including Ethernet routers and switches. Vision is based on a Microsoft Kinect. The robot has two arms with grippers. The software includes Ubuntu GNU/Linux, ROS, and custom code in C++, Python, JAVA, and Lisp.

The cost of the prototype is unknown, but at the time the design was open sourced, the researchers noted that similar robots typically cost as much as 400,000 euros (approximately $523,000 USD) but "our aim is that within a few years you'll be able to build the successor to our AMIGO for 10,000 euro."

No kits or assembled robots are available for purchase at this time.

Read more about AMIGO:
www.techunited.nl/en/amigo
Get the design:
https://robotics.wtb.tue.nl/svn/rop/AMIGO



Ikimo Robot Platform
Date: 2011
Hardware License: CC BY-SA v3.0
Software License: CC BY-SA v3.0
(See note in article about CC licensed software)

The Ikimo robot is a four-wheeled chassis that uses skid-steering. The design includes both the robot's chassis and an Atmel-based microcontroller. Drive motors may be either RC servos or DC motors. It appears there are no motor encoders or other sensors in the base design, but these features could be incorporated by the user. Full parts kits are available for $135 USD or you can buy controller boards and laser-cut your own chassis.


Photo provided by InMojo

Get the design: **http://inmojo.com/store/inmojo-market/item/ikimo-robot-platform/#downlaods**
Buy an Ikimo kit or controller: **http://inmojo.com/store/inmojo-market/item/ikimo-robot-platform/**

Mini-Skybot
Date: 2011
Hardware License: CC BY SA v3.0
Software License: GNU GPL

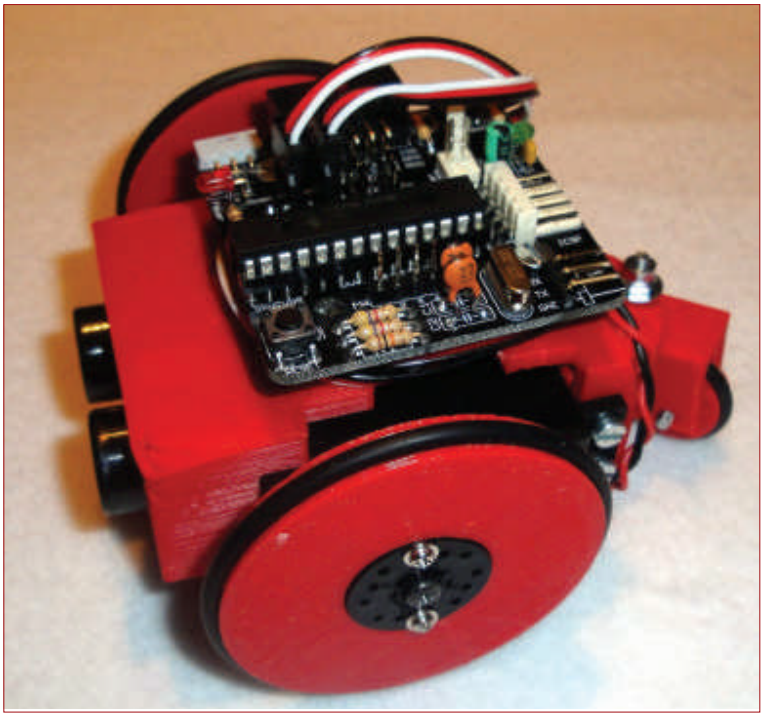


Photo by user IeaRobotics wiki user Obijuan, GNU GDFL 1.2

Mini-Skybot is a differential drive robot with two RC servo-driven wheels and a castor. The robot lacks motor encoders and only has a single sensor — an ultrasonic rangefinder. The robot is designed to be 3D-printable and uses open source mechanics and electronics. Of particular note about the Mini-Skybot is that it was designed using only open source tools. The design was created for educational use. The primary goals were minimal expense and a simple design useful for teaching students programming. The design was a collaboration between the Carlos III University of Madrid Robotics Lab and Universidad Autónoma de Madrid.

The Mini-Skybot Wiki provides access to the hardware design files and software, as well as step-by-step assembly instructions with photos. No kits or assembled units are available at this time. The project is still active and a Miniskybot v2 prototype is being developed.

Get the design: **www.iearobotics.com/wiki/index.php?title=Mini-Skybot**

Thymio II Project
Date: 2011
Hardware License: CC BY SA v3.0
Software License: GNU LGPL

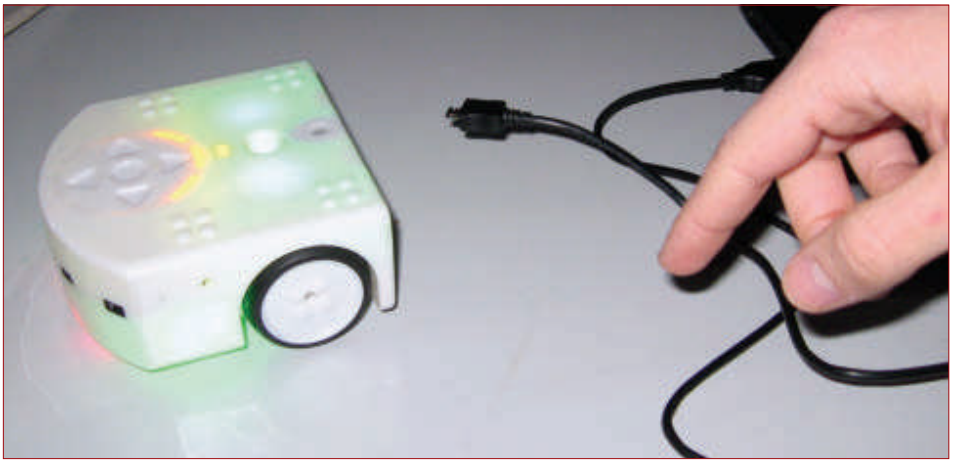


Photo by flickr user colin&claire, CC BY-SA 2.0

Thymio II is an improved version of an earlier robot — not surprisingly called Thymio. The robot was developed as a collaboration between Ecole Polytechnique Fédérale de Lausanne (EPFL) and the Ecole Cantonale d'Art de Lausanne. Goals of the project include minimal cost and wide distribution. Thymio II is a small differential drive robot with a variety of behaviors. It can be programmed using a visual programming language. A fully assembled Thymio II robot can be purchased for $200 USD. Hardware design files including CAD files and schematics can be found on the Thymio II website. No kits are available at this time.

Get the design: **https://aseba.wikidot.com/en:thymio**
Buy a complete Thymio II: **https://techykids.myshopify.com/**

Other Free/Open Hardware Robots?

It's almost certain I missed some projects, and new projects will have started by the time this article is printed. You can find an updated version of this article online at the **FreeIO.org** website that also includes video of each of the robots in action.

Tiny Wanderer
Date: 2011
Hardware License: GNU GPL
Software License: GNU GPL

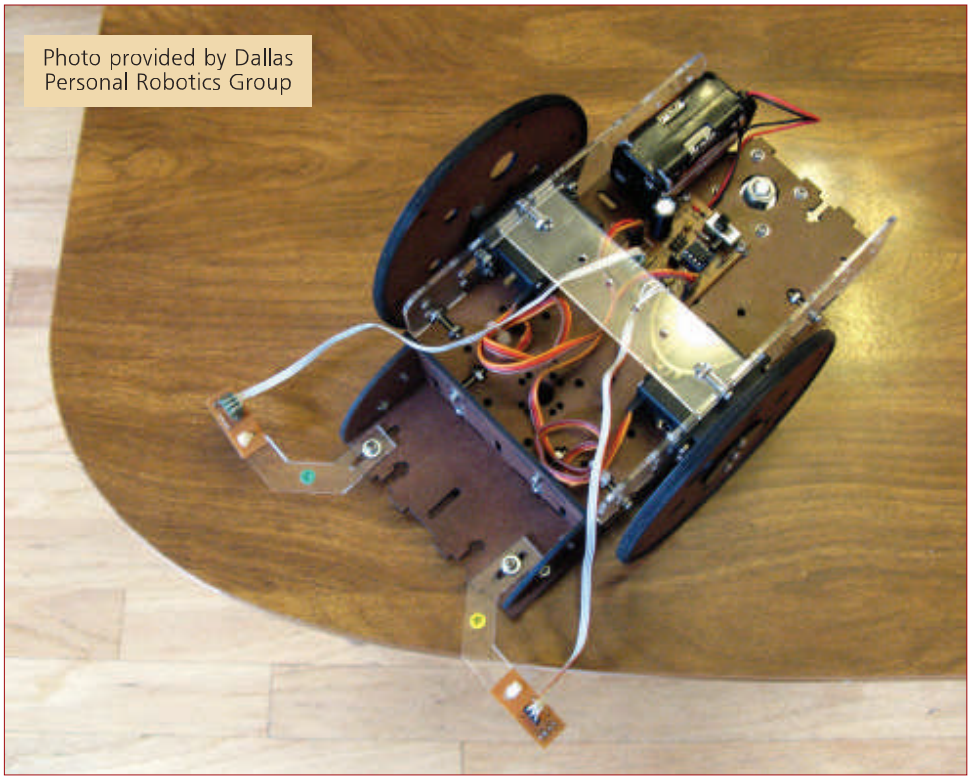


Photo provided by Dallas Personal Robotics Group

The Tiny Wanderer robot is the result of a year-long series of DIY robot workshops at the Dallas Personal Robotics Group that took new users through the process of building an entire robot from scratch. The process included designing and building a PCB for an Atmel AVR microcontroller, laser-cutting a robot chassis, and combining the parts into a small differential drive robot with several sensors for light detection, edge detection, and line following. The Tiny Wanderer robot made the rounds of Instructables, *MAKE Magazine*, and other DIY websites in 2011. Many robots were built in a variety of materials including acrylic and wood. A full kit of parts is available through the Maker SHED for $179 USD. All design files, schematics, and source code are available for download. Design files for laser-cut and scroll saw versions are provided.

Get the design: **http://svn.dprg.org/viewvc/dprg/TinyWanderer/**
Buy a Tiny Wander kit: **www.makershed.com/Tiny_Wanderer_Complete_Kit_p/mstw01.htm**

Aracna Open Source Quadruped Platform
Date: 2012
Hardware License: GNU GPL
Software License: GNU GPL

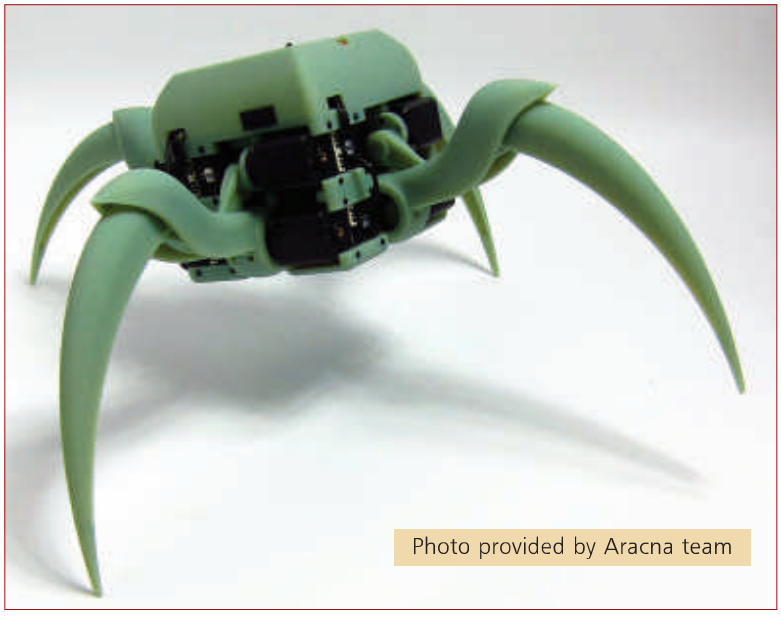


Photo provided by Aracna team

The Aracna Open Source Quadruped Platform is a project of Cornell Creative Machines Lab. The design intentionally uses non-intuitive motor commands for locomotion in order to create a challenge for new gait learning algorithms, developed through evolutionary algorithms. The design uses a combination of off-the-shelf and 3D printed parts. One unique design characteristic is that unlike other legged robots, the legs do not contain any servos. All servos are located in the body and are connected to leg joints through mechanical linkages. A paper is available that provides more technical detail on the design, "*Aracna: An Open-Source Quadruped Platform For Evolutionary Robotics.*" Component costs are estimated to be $1,400 USD. At this time, no kits or assembled units are available for sale.

Read the technical paper: **http://goo.gl/k7HTZ**
Get the design: **http://creativemachines.cornell.edu/aracna**

ArcBotics Hexy
Date: 2012
Hardware License: CC BY-SA v3.0
Software License: MIT

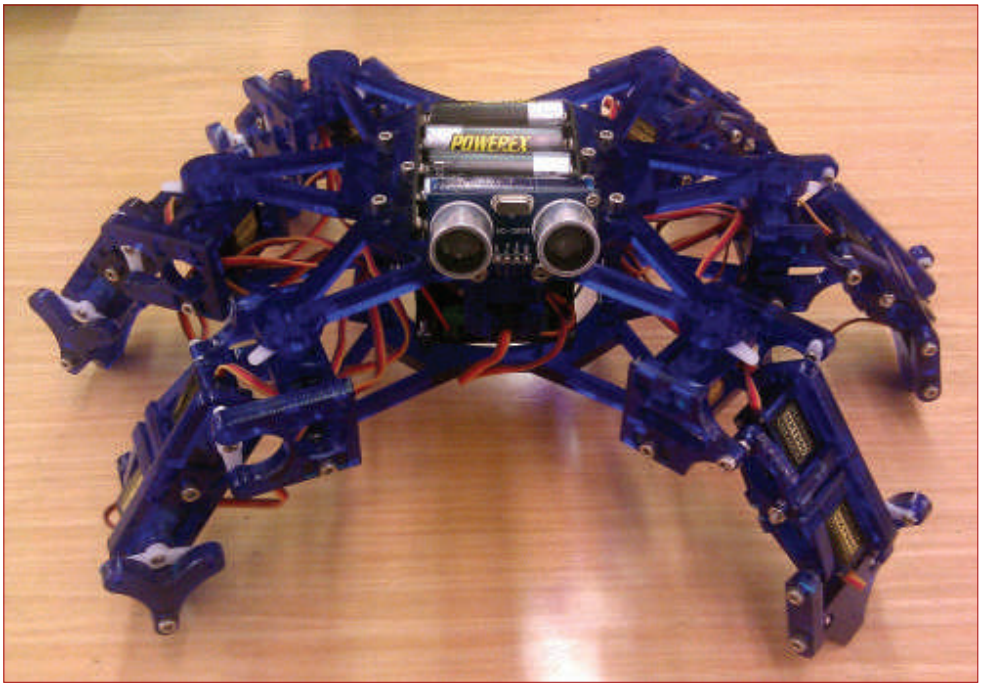


Photo by ArcBotics, CC BY-SA v2.0

The ArcBotics Hexy is a hexapod robot. The design was funded by a Kickstarter campaign in 2012. The robot uses 19 RC servos — three for each of the six legs, plus one for a sonar-equipped sensor array. The design aims at being low cost and the designers claim a completed Hexy costs only 1/10 the price of proprietary hexapod kits. Structural components are laser-cut acrylic. The controller — known as the Servotor32 — is also an open source design. Software and hardware design files are available on GitHub for those who like to build from scratch.A complete Hexy kit is available for $250 USD. A fully assembled Hexy robot is available for $500 USD.

Get the design: **https://github.com/ArcBotics/Hexy**
Buy a Hexy kit: **https://wepay.com/stores/arcbotics/item/hexy-the-hexapod-robot-962355**
Buy a complete Hexy: **https://wepay.com/stores/arcbotics/item/hexy-the-hexapod-robot-fully-assembled-653710**

OpenROV
Date: 2012
Hardware License: CC BY-SA v3.0
Software License: CC BY-SA v3.0
(See note in article about CC licensed software)



Photo by flickr user esabet, CC BY 2.0

The OpenROV project has developed a hardware design for an ROV capable of reaching depths of 100 meters. You can build the design yourself from scratch using the provided files, or you can buy kits or completely assembled ROVs.

The design relies on the BeagleBone microcontroller — itself a partially free hardware design.

Get the design: **http://wiki.openrov.com/index.php/Main_Page**
Buy and OpenROV kit: **http://openrov.myshopify.com/collections/frontpage/products/openrov-kit**

Open Hardware Mobile Manipulator (OHMM)
Date: 2012
Hardware License: CC BY SA v3.0
Software License: GNU GPL

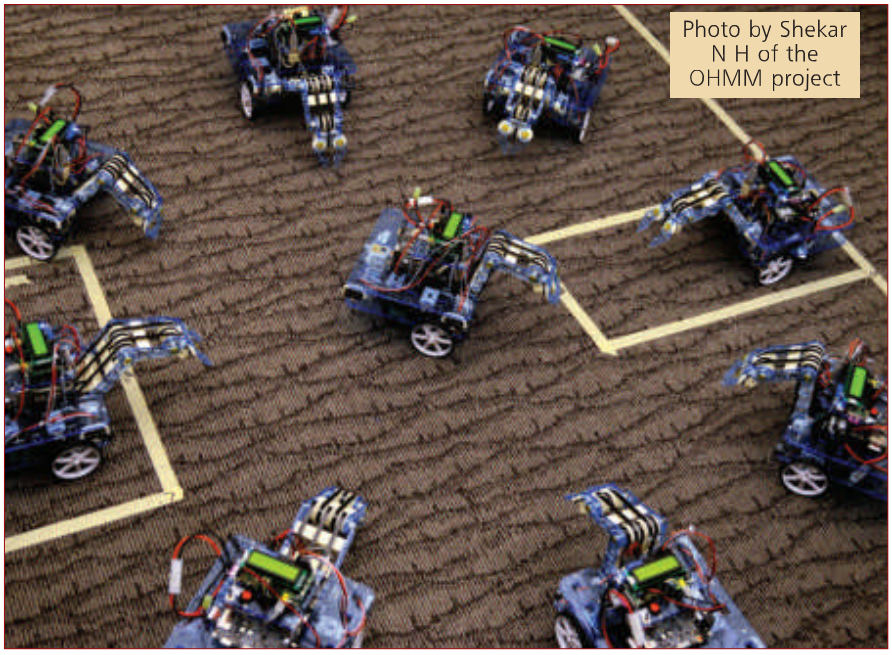


Photo by Shekar N H of the OHMM project

The Open Hardware Mobile Manipulator project was started by the Geometric and Physical Computing group at Northeastern University College of Computer and Information Science. The group set out to design a small mobile robot with a manipulator arm for educational use. A design goal was to use off-the-shelf hardware and software as much as possible. Pololu Orangutan SVP and Pandaboard microcontrollers are used. There are no custom PCBs. CAD files are provided for creating a variety of 3D printed and laser-cut components. Once all of the parts are available, the only tools needed to assemble them are pliers and a screwdriver. Custom software was developed which integrates a number of existing free software robotics libraries such as OpenCV, JavaCV, and libfreenect. Current work is underway to develop an ROS interface. The total build cost is estimated at $1,730 USD. No kits or assembled robots are available at this time.

Get the design: **www.ohmmbot.org**

DIY NEVER HAD A BETTER FRIEND...

Take your DIY project to the next level with PanaVise work holding tools. Perfect for hobbies, soldering, electronics, crafting & repair, PanaVise tools have hundreds of uses only limited by your imagination.

PanaVise, "better than a third hand!"

PANAVISE®
Innovative Holding Solutions

Now available at RadioShack.

Model 301
640-0178

Model 324
640-0177

www.radioshack.com/StoreLocator

Robot MESH
www.robotmesh.com
FREE SHIPPING
SCHOOL POs ACCEPTED

Robotic Kits and Controllers for All

VEX · iROBOT · MINDSTORMS · fischertechnik

Also:
Flowol 4

Drag-Drop Flowchart Programming for all the above robotics controllers

PC & Mac

Includes *mimic* simulations with curriculum



30-day free trial: www.flowol.com

Interface and Communication Techniques to Control the World

(or, at least H-bridges)

by Fred Eady

Are you in control? Or, do you just think you are? There are so many gadgets for us to control out there these days. Do you find yourself writing massive or complex applications for every little thing you want to rule? Many times, we overlook the easy way to get at those remote devices. And, many times we fail to play the hand we're dealt when it comes to interfacing with a particular device.

Knowing the fundamentals of what we are remotely targeting is a must. Unless you're monitoring the pitch and yaw of a supersonic fighter jet in parts unknown, communicating with and interfacing to most everyday gadgets with today's commercial microcontrollers is a walk in the park.

This month, we're going to examine some basic interface and communications techniques. If we can, we'll build on them to make them more powerful without adding exponential complexity.

What Do You Really Need?

That depends. Will your target be physically accessible to your control or monitoring device? Is a data radio involved with your project? Do you have the firmware tools to support your application? Will you need a particular piece of test equipment? One thing is for sure. You will need to bring enough microcontroller to the fray.

Such a microcontroller platform is shown in **Photo 1**. The microcontroller can be scaled to fit the application's needs. Any microcontroller that is pin-compatible with the Microchip PIC18F47J13 will work. You have the option to change the crystal frequency or use no crystal at all. If your application requires time keeping, pads are there for a 32.768 kHz crystal complex. Take a look at **Schematic 1** and you'll see that the serial port can be converted to a USB port by employing the services of the onboard FT232RL.

Again, if you don't need USB and need a standard serial port, leave the USB-to-UART bridge at home. Power for this PIC-based platform can be had by way of the USB portal, a battery, or a wall wart. Input voltages up to a maximum of 16 VDC can be accommodated. The debugging/programming port is part of the printed circuit board (PCB) base design. An inexpensive PICkit 3 is targeted as the default programmer/debugger. The base PCB design also includes holes for mounting male or

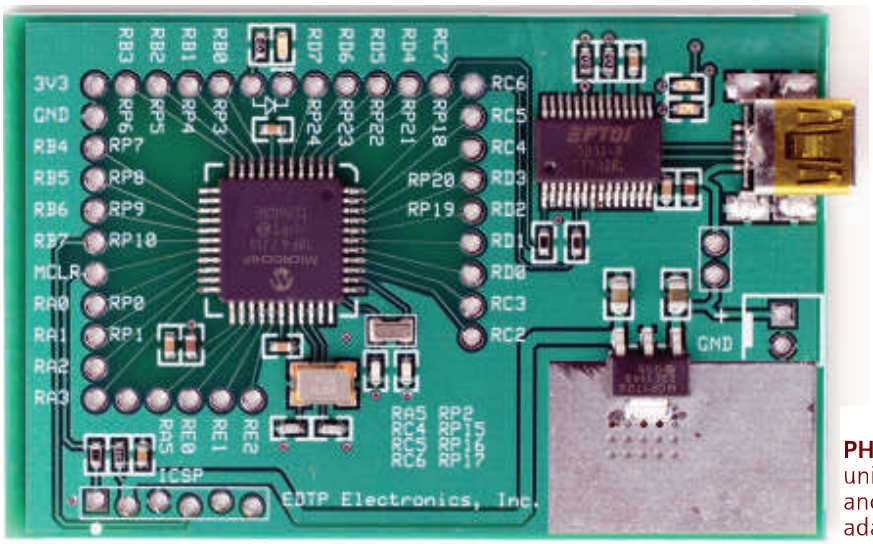
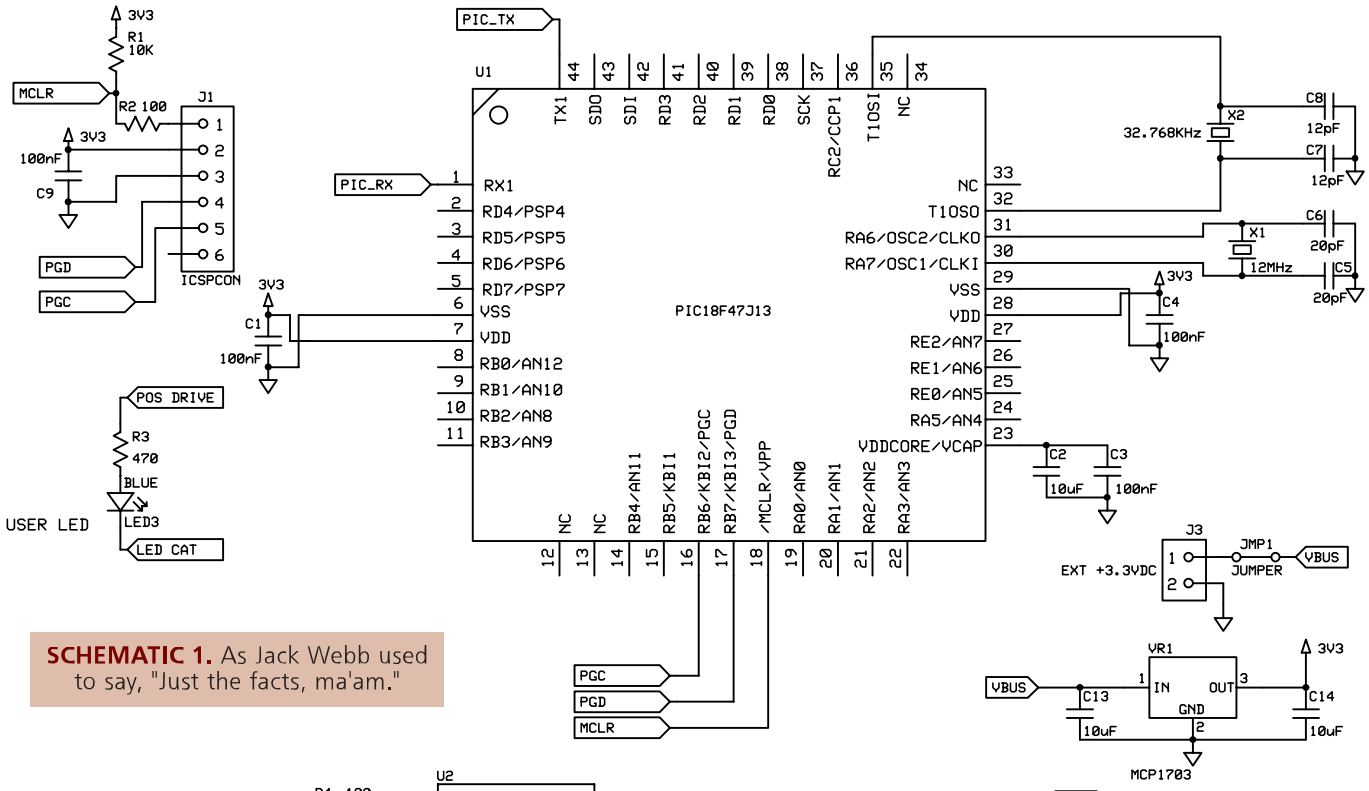
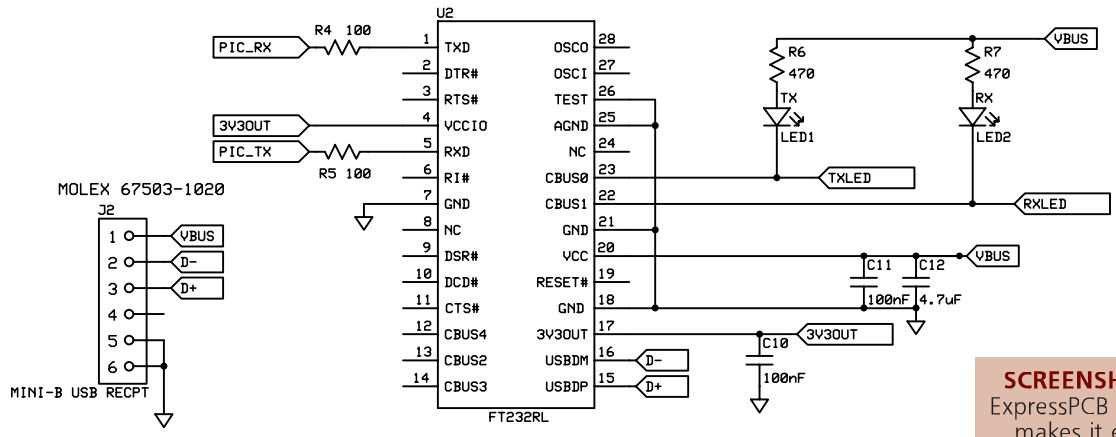


PHOTO 1. This simple little board is actually a universal soldier in disguise. With some tricky coding and hardware peripheral manipulation, it can be adapted to most any application.

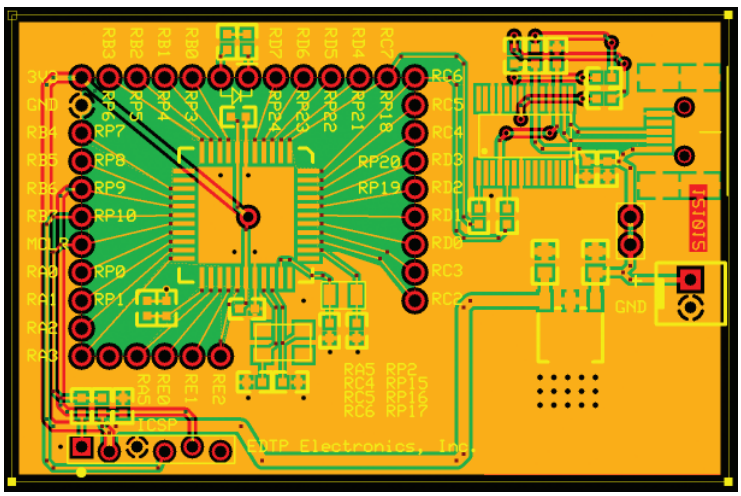


SCHEMATIC 1. As Jack Webb used to say, "Just the facts, ma'am."



SCREENSHOT 1. Having the master ExpressPCB board and schematic files makes it easy to duplicate the base design or customize it for your particular application.

female 0.1 inch pitch headers. The PIC can be programmed using assembler, PICBASIC PRO, CCS C, and Microchip C to name just a few. In the case of the PIC18F47J13, some of the GPIO pins are relocatable. This little board provides lots of bang for the buck, as well. **Screenshot 1** is a graphical representation of the PIC18F47J13 board we have been discussing. I'll supply the ExpressPCB files if you want to build one of these up on your own. You can get component guidance by viewing **Schematic 1**.



Okay ... I Need That

So, you want to know how to use it. One of the most useful hardware tools we have at our disposal is the venerable H-bridge. H-bridges are interesting

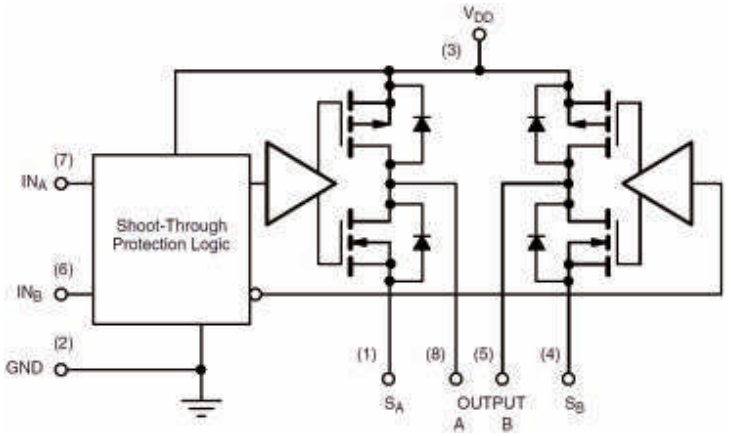


FIGURE 1. All of the hard stuff about implementing H-bridges is taken care of in the input section of the Si9986.

devices. You can build them from scratch or obtain them prepackaged. They come in high current versions or low power, low current versions. Their ability to place opposite polarities on the attached load offers a ton of functional possibilities. A good example of an H-bridge that is well suited for low power applications is the Si9986. The Si9986 is able to switch up to 12 VDC with a continuous load of 1A. The eight-pin Si9986 is also suitable for PWM (pulse width modulation) applications as it can switch at frequencies up to 200 kHz. A block diagram of the Si9986 is constructed in **Figure 1**.

TRUTH TABLE			
IN_A	IN_B	OUT_A	OUT_B
1	0	1	0
0	1	0	1
0	0	0	0
1	1	HiZ	HiZ

FIGURE 2. You can equate the Si9986's four output states to motor terms such as forward, reverse, brake, and free-wheeling.

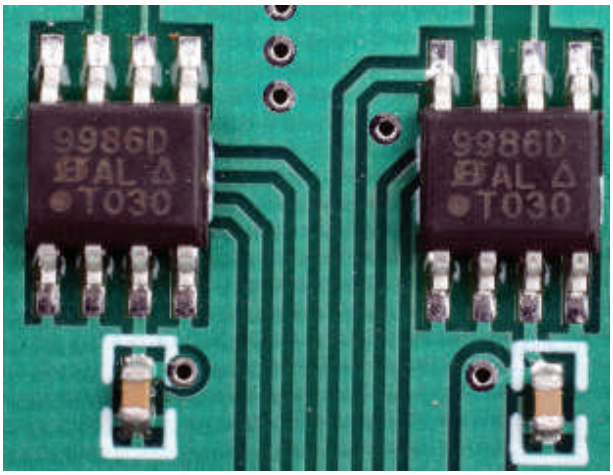
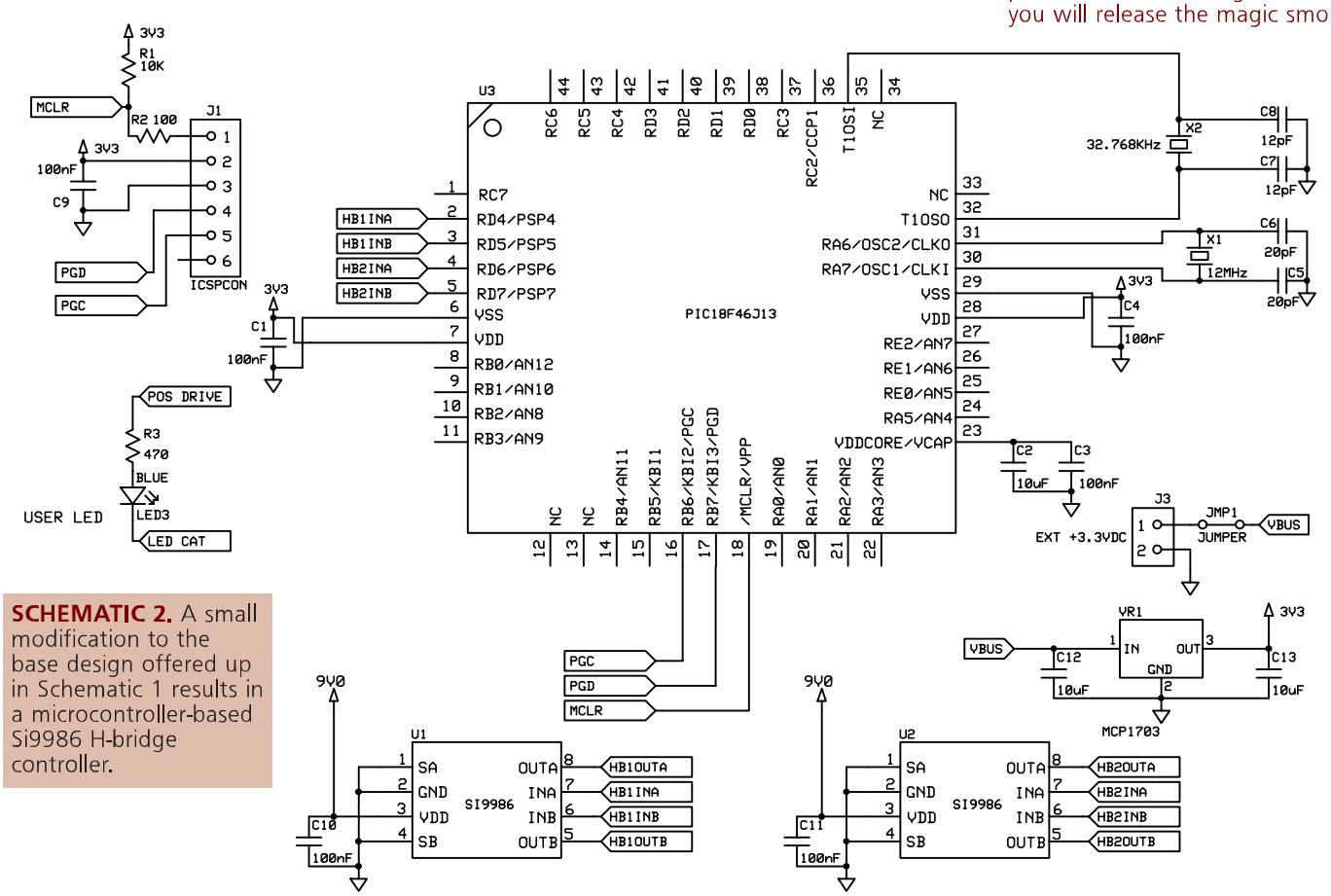


PHOTO 2. You can assemble a basic H-bridge from discrete parts that will probably cost less. However, if your code fails to switch the active components of the H-bridge correctly, you will release the magic smoke.



SCHEMATIC 2. A small modification to the base design offered up in Schematic 1 results in a microcontroller-based Si9986 H-bridge controller.

The Si9986 is easy to interface as its inputs are TTL compatible. So, combinatorial logic resulting from TTL compatible logic gates could be employed to control the Si9986's outputs. There are two inputs. That means the Si9986 can assume one of four possible states. I've outlined the states in **Figure 2**.

I applied the Si9986 in a real world application using the base circuit laid out in **Schematic 1**. However, instead of using the PIC18F47J13, the application hardware you see in **Schematic 2** employed the PIC18F46J13. The pair of Si9986s under the lights in **Photo 2** is the silicon equivalent to the paper tigers depicted in **Schematic 2.** How easy is it to write Si9986 drivers for the PIC18F46J13? Watch this:

```
//*****************************************
//*    H-BRIDGE DEFINITIONS
//*****************************************
#define    HB1INA      LATD,4
#define    HB1INB      LATD,5
#define    HB2INA      LATD,6
#define    HB2INB      LATD,7
```

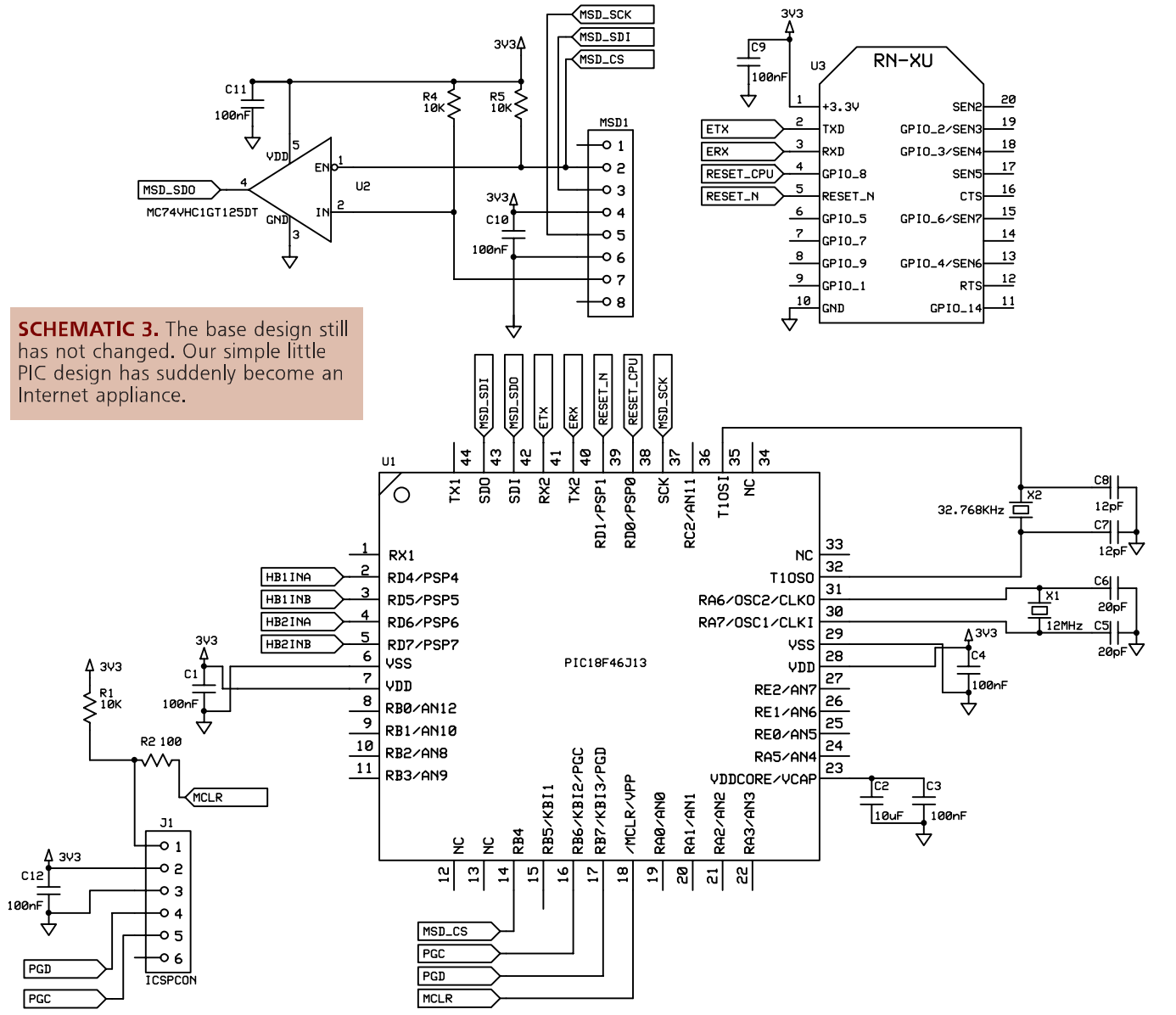
That does it for the input aliases. Now, let's write some C macro code to enable the pair of Si9986 H-bridges to fall into any of the available output states:

```
#define hb1brake      bit_clear(HB1INA);    \
                      bit_clear(HB1INB);
#define hb2brake      bit_clear(HB2INA);    \
                      bit_clear(HB2INB);

#define hb1outa       bit_set(HB1INA);      \
                      bit_clear(HB1INB);
#define hb2outa       bit_set(HB2INA);      \
                      bit_clear(HB2INB);

#define hb1outb       bit_clear(HB1INA);    \
                      bit_set(HB1INB);
#define hb2outb       bit_clear(HB2INA);    \
                      bit_set(HB2INB);

#define hb1open       bit_set(HB1INA);      \
                      bit_set(HB1INB);
#defineh b2open       bit_set(HB2INA);      \
                      bit_set(HB2INB);
```



SCHEMATIC 3. The base design still has not changed. Our simple little PIC design has suddenly become an Internet appliance.

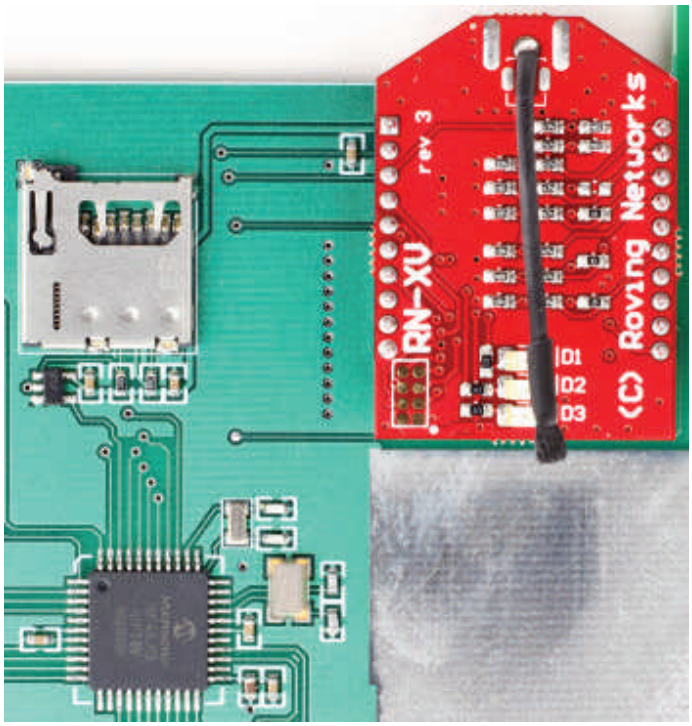


PHOTO 3. The RN-171-XV is very easy to configure and contains its own TCP/IP stack. Once the TCP connection is established, data flows over the Wi-Fi connection via the RN-171-XV's serial port.

The *bit_set* and *bit_clear* macro statements are coded in CCS C compiler syntax. The LATx definitions are foreign to the CCS compiler unless you inform it where to find the LATx registers:

```
#byte   LATE = 0x0F8D
#byte   LATD = 0x0F8C
#byte   LATC = 0x0F8B
#byte   LATB = 0x0F8A
#byte   LATA = 0x0F89
```

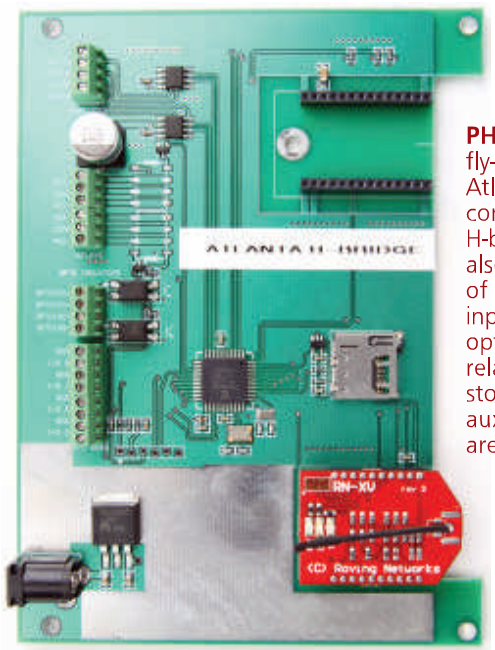


PHOTO 4. This is a fly-over view of the Atlanta H-bridge controller. The H-bridge controller also features a pair of optically isolated inputs, two optional DPDT relays, microSD storage, and an auxiliary device area.

What you don't see in **Photo 2** are the screw terminals that take the Si9986 H-bridge's output pin states to the loads. We now have the power to switch external device power, drive motors, open and close valves, and actuate solenoids with minimal code and minimal hardware. For instance, to drive OUTPUTA of H-bridge 1, we would simply code:

```
hb1outa;
```

I'm in Florida ... the H-bridge is in Georgia

No worries. We can flip that H-bridge in Atlanta from Key West if we have to. All we need to do is add some additional hardware capability to our initial design. Even with the additional parts, the base design we started with in **Schematic 1** will not radically change.

In the good old days, reaching out to the H-bridge in Atlanta would have entailed the installation of a modem at both ends of the communications link. Most likely, the modems would be dial-up types since leased telephone lines were expensive and the H-bridge would probably not need immediate around-the-clock service. Today, if there is a civilized human being nearby, you will most likely find an Internet portal. So, all we need to do is add some hardware that can give the PIC18F46J13s at each end access to the Internet. Once that's done, we can choose to communicate elegantly with a custom application or use free readily available communications resources.

The hardware design you see in **Schematic 3** wasn't pulled out of the air. It is a solid design that has worked for me many times in the past. The first order of business is to get the RN-171-XV Wi-Fi radio on the Florida LAN. The Key West RN-171-XV needs to be configured with the following:

- Gateway address
- IP address
- Local port number
- Channel
- Authorization type
- Authorization key or pass phrase
- LAN SSID

The Key West IP and WLAN setups are shown in **Screenshot 2**. DHCP is disabled to allow for a stable local IP address that can always be matched to the local port address of 9000. The gateway address is the local IP address of the router or access point that interfaces the LAN to the Internet. I turn on both TCP and UDP so as not to miss anything that may use UDP as a communications datagram carrier.

In that the Key West router/access point requires WPA2 authorization, we must configure the Key West RN-171-XV with a pass phrase. The Key West SSID value is important, as well. The Join = 1 instructs the Key West RN-171-XV to only join a LAN whose SSID is *keywest*. The Key West LAN will always operate on channel 11, and that is reflected in the Key West RN-171-XV setup. Both ends of the link need to belong to their respective resident LAN. So, what was done for the Key West RN-171-XV must also

be done for the Atlanta RN-171-XV.

Since there are no static IP addresses at either end of the link and the 192.168.0.xxx addresses are not routable on the Internet, we must have a way to contact the Atlanta router from Key West using a routable IP address. The problem is that the Atlanta router's IP address is apt to be changed at any time by the ISP (Internet Service Provider) servicing the Atlanta router's Internet connection. So, to make sure we can always find the Atlanta router, we'll set up a DynDNS (Dynamic DNS) entry for it. Then, all we need to know is Atlanta's local port number which is configured in the Atlanta router and the Atlanta RN-171-XV. That port number happens to be 8000.

DynDNS is a billable service ($20 a year) that allows the association of an Internet device IP address with a unique host name. If the host router IP address changes, DynDNS sees that and associates the Atlanta router's new IP address with the respective host name. For instance, the Atlanta H-bridge controller's DynDNS host name is **atlanta hbridge.DynDNS-remote.com**. No matter what IP address the Atlanta router adopts, the IP address will always be associated with the DynDNS host name. So, to contact the Atlanta router, all we need to do is enter the DynDNS host name we set up on the DynDNS website (*atlantahbridge.DynDNS-remote.com*), followed by the local port number assigned to the Atlanta RN-171-XV (8000).

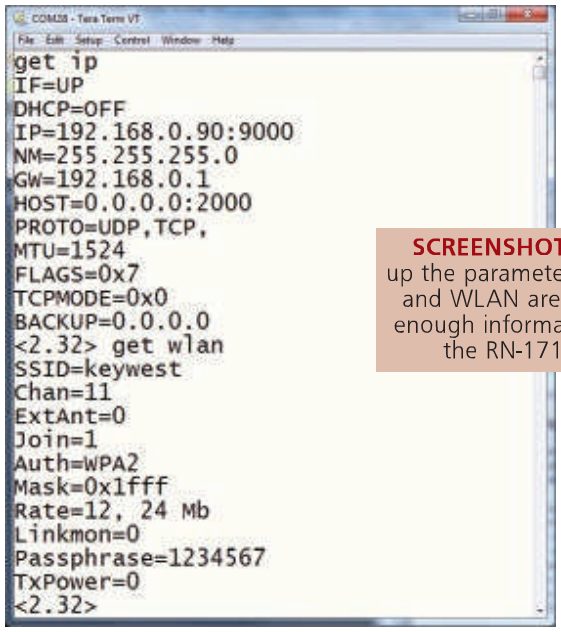
The Atlanta and Key West routers must be capable of supporting the DynDNS service. Most all of the current routers on the market today support this service.

Florida-Georgia Dry Run

We can test the feasibility of our remote H-bridge application right here on the bench. The PIC18F46J13-based Atlanta node is pictured in **Photo 4**. We are primarily interested in the H-bridges. However, the Atlanta H-bridge controller can also provide remote access to a pair of DPDT relays, two optically isolated inputs, several digital I/O pins, microSD storage, and an auxiliary device socket. The object of this test session is to contact the Atlanta H-bridge node and issue a simple command to activate and deactivate an H-bridge output. First, we must load the Atlanta node with the appropriate receiver code:

```
#INT_RDA2
void  RDA2_isr(void)
{
  unsigned char data,i;
  unsigned char tmphead;
  data = RCREG2;     //read the received data
  tmphead = ( USART_RxHead + 1 ) &
  USART_RX_BUFFER_MASK;
  USART_RxHead = tmphead;   //store new index

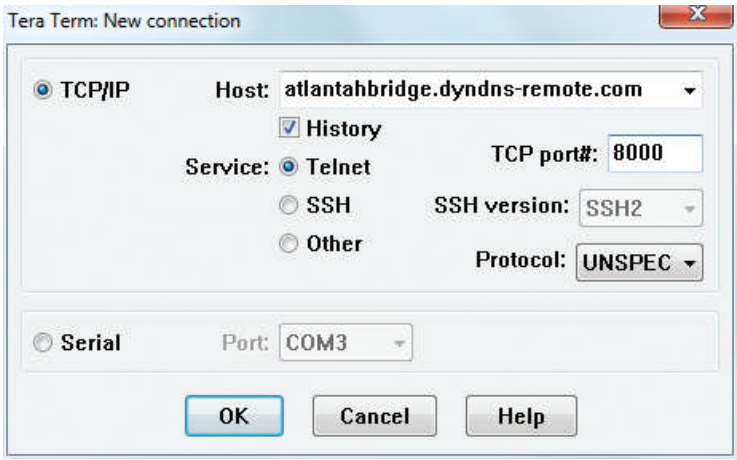
  if ( tmphead == USART_RxTail )
  {
     //ERROR HAS OCCURRED - RESET THE RECEIVER
     //bit_clear(CREN2);
     //bit_set(CREN2);
  }
  USART_RxBuf[tmphead] = data;
       //store received data in buffer
}
```



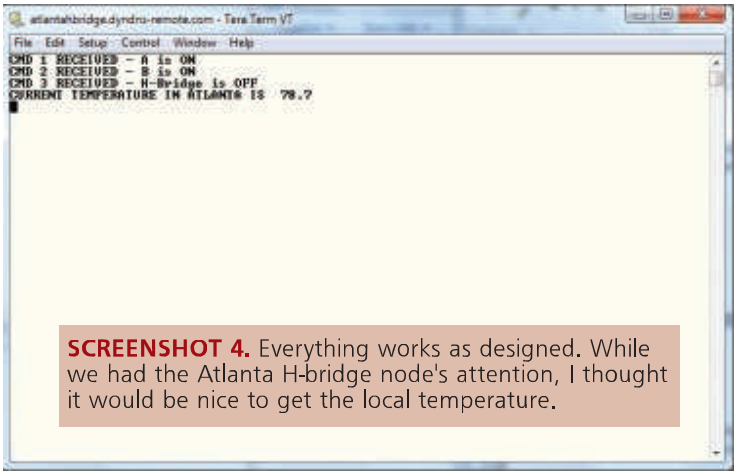
SCREENSHOT 2. Setting up the parameters in the IP and WLAN areas provides enough information to get the RN-171-XV online.

```
//************************************************
//*     USART Receive Character Function
//************************************************
unsigned char recvchar(void)
{
  unsigned char tmptail;

  while ( USART_RxHead == USART_RxTail );
       // wait for incoming data
  tmptail = ( USART_RxTail + 1 ) &
  USART_RX_BUFFER_MASK;
       // calculate buffer index
  USART_RxTail = tmptail;
       // store new index
  return USART_RxBuf[tmptail];
       // return data
}
//************************************************
//*     USART Character Waiting Function
//************************************************
unsigned char CharInQueue(void)
{
  return(USART_RxHead != USART_RxTail);
}
```



SCREENSHOT 3. Tera Term is a luxury here. We can also use the Windows or Android Telnet applications to contact the Atlanta H-bridge node. In fact, anything that can run as a Telnet client can access the Atlanta H-bridge node.



SCREENSHOT 4. Everything works as designed. While we had the Atlanta H-bridge node's attention, I thought it would be nice to get the local temperature.

The Atlanta node receiver modules respond to an interrupt that is triggered by an incoming character. The incoming data is retrieved from the PIC18F46J13's UART and placed in the receive buffer. The presence of any data in the buffer is detected by the main program by calling the *CharInQueue* function. If the receive buffer contains a character, the *recvchar* function is called to remove the character from the buffer. The retrieved character is then processed by the main program. In the case of this test, the incoming character will be an H-bridge command. The main test routine code looks like this:

```
if(CharInQueue())
{
     bitein = recvchar();
     switch(bitein)
     {
 case '1':
     hb1outa;
     fprintf(atlanta,"CMD 1 RECEIVED - A is
     ON\r\n");
                  break;
                  case '2':
                     hb1outb;
     fprintf(atlanta,"CMD 2 RECEIVED -
     B is ON\r\n");
break;
case '3':
     hb1brake;
     fprintf(atlanta,"CMD 3 RECEIVED -
     H-Bridge is OFF\r\n");
break;

     }
}
```

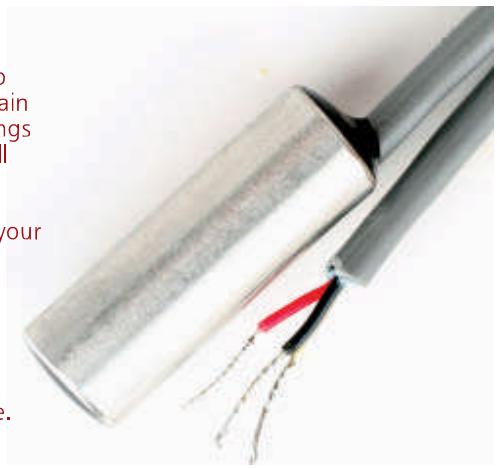


PHOTO 5. You too can accurately obtain temperature readings with this puppy. All you have to do is power it with 3.3 VDC, read it with your microcontroller's analog-to-digital converter, and use the firmware I've provided to extrapolate the actual temperature.

Everything is now in place to test the Atlanta node. We will use Tera Term to contact the H-bridge controller and exercise H-bridge 1. We are sitting on the launch pad in **Screenshot 3**.

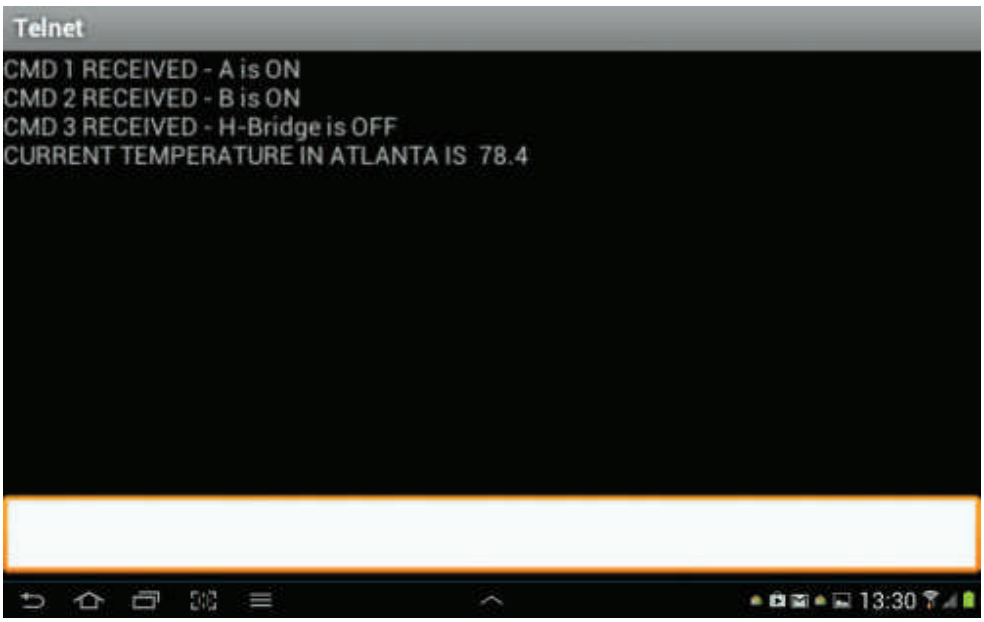
Houston, This is Cape

We have cleared the tower. As you can see in **Screenshot 4**, all three commands that were issued from the Tera Term Telnet session were acknowledged by the Atlanta H-bridge Internet appliance. What's up with that temperature report?

Since we were in contact with the Atlanta H-bridge node, it would only take a small bit of additional code to collect the local temperature:

```
void getTemp(void)
{
 int1 done;

 set_adc_channel(6);
 delay_ms(10);
  read_adc(ADC_START_ONLY);
  done = adc_done();
  while(!done)
  {
   done = adc_done();
  }
   tempmv = read_adc();
  tempmv *= 0.805664;
  tempC = 0.0512 * tempmv
  - 20.5128;
```



SCREENSHOT 5. This is an Android view of the Atlanta H-bridge Telnet session. This screenshot is presented to you courtesy of my Samsung Galaxy Tab.

```
    tempF = (1.8 * tempC) + 32;
    leftdigit = tempF;
    rightdigit = (tempF * 10) - (leftdigit * 10);
    fprintf(atlanta,"CURRENT TEMPERATURE IN
    ATLANTA IS %3lu.%1lu\r\n",leftdigit,right
    digit);
}
```

All that's left to do is add the command code to fetch the temperature:

```
if(CharInQueue())
{
    bitein = recvchar();
    switch(bitein)
    {
   case '1':
    hblouta;
    fprintf(atlanta,"CMD 1
    RECEIVED - A is ON\r\n");
   break;
   case '2':
    hbloutb;
    fprintf(atlanta,"CMD 2
    RECEIVED - B is ON\r\n");
   break;
   case '3':
    hb2brake;
    fprintf(atlanta,"CMD 3
    RECEIVED - H-Bridge is
    OFF\r\n");
   break;
   case '4':
    getTemp();
   break;
}
```

The temperature probe I used is shown in **Photo 5**. You can get your own ENV-TMP from Atlas Scientific. The ENV-TMP draws practically no current. So, you can actually power it from one of the PIC18F46J13's I/O pins.

We Have Reached Escape Velocity

Now that you know how to use a PIC18F46J13 microcontroller and an RN-171-XV to control devices over a LAN and the Internet, the possibilities are nebulous. And to think, this all started with a very simple PIC circuit which didn't morph as the project requirements increased.

I figure you want to see this work on an Android device. So, I'll leave you with **Screenshot 5**. SV

Sources

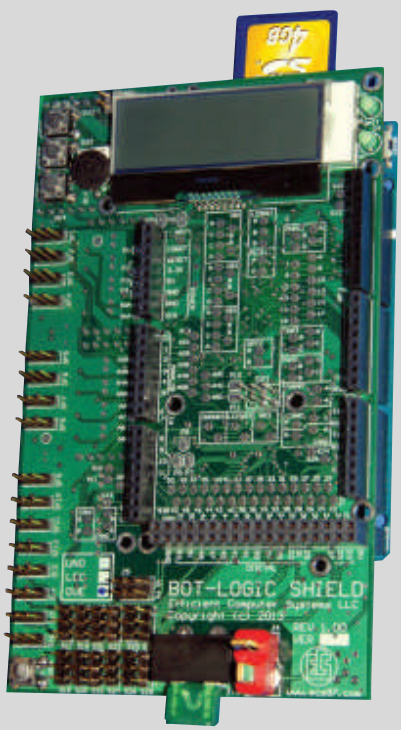
Microchip
PIC18F46J13
PIC18F47J13
RN-171-XV
www.microchip.com

Custom Computer Services
CCS C Compiler
www.ccsinfo.com

DYN
DynDNS
www.dyndns.com

Atlas Scientific
ENV-TMP
www.atlas-scientific.com

30 amp Robot Shield for Arduino



Features:

- Supports 8 – 28 servos
- Servo load measurement on up to 12 servos allowing pressure control and touch detection
- Run Arduino on 3-18 VDC
- Designed to run on a single 4.8 – 7.2 Volt battery pack
- Accepts up to 3 expansion boards and additional shields
- Built-in 3-axis accelerometer
- 3 LED drivers up to 500 ma per output
- Audio output with 1 watt speaker driver

Compatible with Arduino Uno, Leonardo, Mega, and Due

BOT-LOGIC.COM
FAST TRACK ELECTRONICS For Robot Builders
Efficient Computer Systems, LLC • www.bot-logic.com

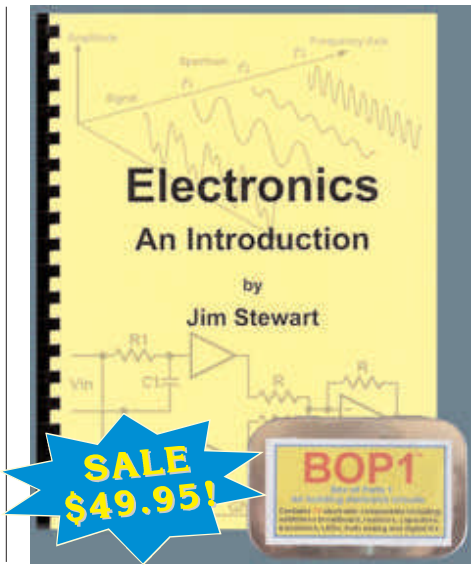
The SERVO Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS

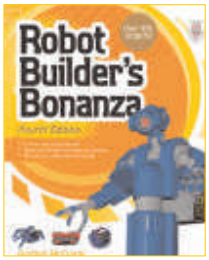


06.2013 FEATURED



ROBOTICS

Robot Builder's Bonanza, Fourth Edition
by Gordon McComb



Robot Builder's Bonanza, Fourth Edition includes step-by-step plans for a number of motorized platforms. The book is described as a compendium of robotics topics, containing more than 100 projects, including 10 robot designs new to the fourth edition. These modular robots are low cost, and are made to be reproduced by readers with no training in mechanical construction.
$29.95*

Mechanisms and Mechanical Devices Sourcebook 5th Edition
by Neil Sclater



Fully revised throughout, this abundantly illustrated reference describes proven mechanisms and mechanical devices. Each illustration represents a design concept that can easily be recycled for use in new or modified mechanical, electromechanical, or mechatronic products. Tutorials on the basics of mechanisms and motion control systems introduce you to those subjects or act as a refresher.
Reg $89.95 Sale Price $79.95

Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists
by Dustyn Roberts



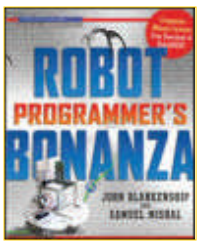
In *Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists*, you'll learn how to successfully build moving mechanisms through non-technical explanations, examples, and do-it-yourself projects — from kinetic art installations to creative toys to energy-harvesting devices. Photographs, illustrations, screenshots, and images of 3D models are included for each project.
$29.95*

Build Your Own Humanoid Robots
by Karl Williams
GREAT 'DROIDS, INDEED!



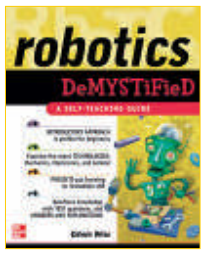
This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than $300. Together, they form the essential ingredients for making your own humanoid robot.
$24.95*

Robot Programmer's Bonanza
by John Blankenship, Samuel Mishal



The first hands-on programming guide for today's robot hobbyist!
Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.
$29.95

Robotics Demystified
by Edwin Wise



YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!
Now anyone with an interest in robotics can gain a deeper understanding — without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **$19.95**

We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Save $10.00
Only $57.95!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, 09, 10, 11 and 2012.

Any bot builders out there? Get cool robotics stuff from my store!

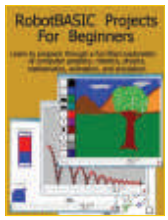
Call me at my order desk!

Visit my online store @ www.servomagazine.com



RobotBASIC Projects For Beginners

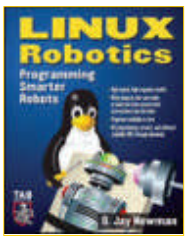
by John Blankenship, Samuel Mishal



If you want to learn how to program, this is the book for you. Most texts on programming offer dry, boring examples that are difficult to follow. In this book, a wide variety of interesting and relevant subjects are explored using a problem-solving methodology that develops logical thinking skills while making learning fun. RobotBASIC is an easy-to-use computer language available for any Windows-based PC and is used throughout the text.
Price $14.95

Linux Robotics

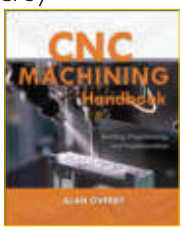
by D. Jay Newman



If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.
$38.95

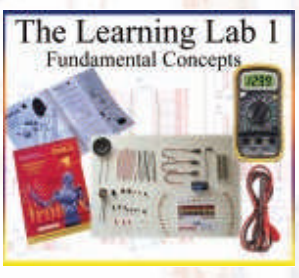
CNC Machining Handbook: Building, Programming, and Implementation

by Alan Overby



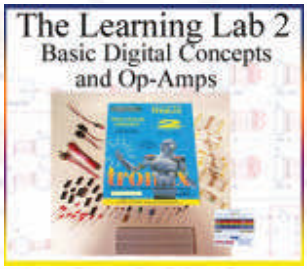
The *CNC Machining Handbook* describes the steps involved in building a CNC machine and successfully implementing it in a real world application. Helpful photos and illustrations are featured throughout. Whether you're a student, hobbyist, or business owner looking to move from a manual manufacturing process to the accuracy and repeatability of what CNC has to offer, you'll benefit from the in-depth information in this comprehensive resource. $34.95

FOR BEGINNER BOT BUILDERS



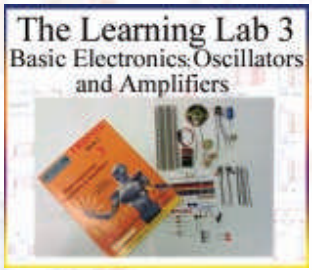
The Learning Lab 1
Fundamental Concepts

$59.95



The Learning Lab 2
Basic Digital Concepts and Op-Amps

$49.95



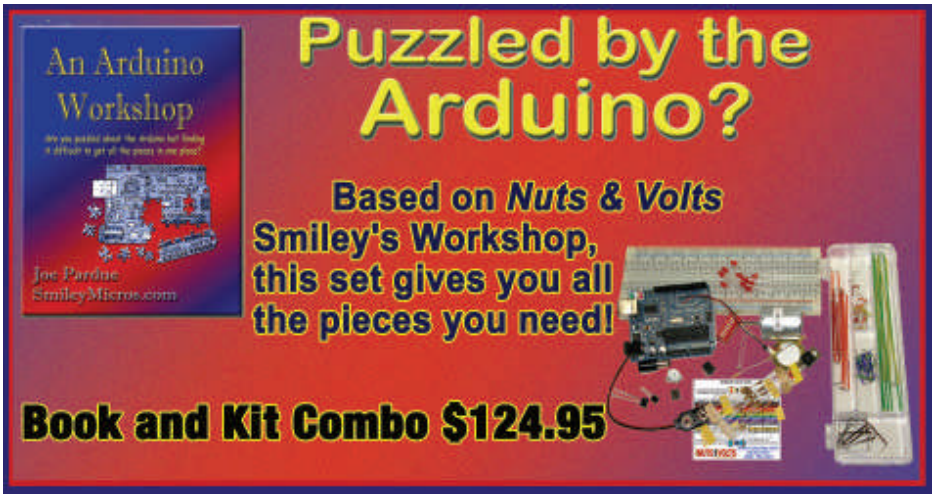
The Learning Lab 3
Basic Electronics: Oscillators and Amplifiers

$39.95

The labs in this series — from GSS Tech Ed — show simple and interesting experiments and lessons, all done on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.
For more info and a promotional video, please visit our webstore.

Or order online
www.servomagazine.com

SPECIAL OFFERS



Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley



Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse.

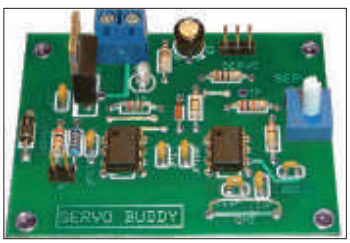
Reg $24.95 Sale Price $19.95





PROJECTS

The SERVO Buddy Kit



An inexpensive circuit you can build to control a servo without a microcontroller.



For more information, please check out the May 2008 issue or go to the SERVO webstore.

Includes an article reprint.
Subscriber's Price **$39.55**
Non-Subscriber's Price **$43.95**

3D LED Cube Kit

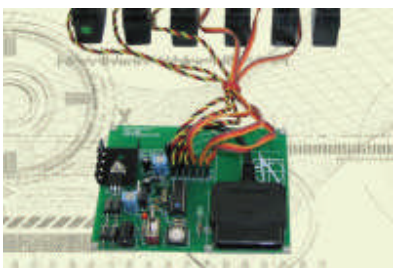


From the article "Build the 3D LED Matrix Cube" as seen in the August 2011 issue of *Nuts & Volts Magazine*.

This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes. Colors available: Green, Red, Yellow & Blue. Jig and plastic cases also available.

Subscriber's Price **$57.95**
Non-Subscriber's Price **$59.95**

PS2 Servomotor Controller Kit



This kit accompanied with your own PlayStation controller will allow you to control up to six servomotors. Includes all components and instruction manual.



For more information, please see the February 2011 edition of *SERVO Magazine*.

Assembled units available!
Subscriber's Price
$79.95
Non-Subscriber's Price
$84.95

ROBOTSHOP WILL INTRODUCE...

SERVO ERECTOR SET V1.1

05.24.2013

GET READY TO CHANGE THE WORLD OF ROBOTICS

Lynxmotion

Imagine it. Build it. Control it.™

THE COUNTDOWN HAS BEGUN

Lynxmotion is a trademark of
RobotShop Distribution inc.

RobotShop.com
Robotics at Your Service!™



Curiosity and the Self-Driving Car

by Paul Bouchier and Jason Henriksen

In June 2012, the Dallas Personal Robotics Group (DPRG) announced that its fall Roborama contest would be a 400 foot racetrack-style course, held outdoors with rough terrain. The course was not precisely specified, but it would have a hairpin turn, left and right turns, and its edges would be marked with small red flags.

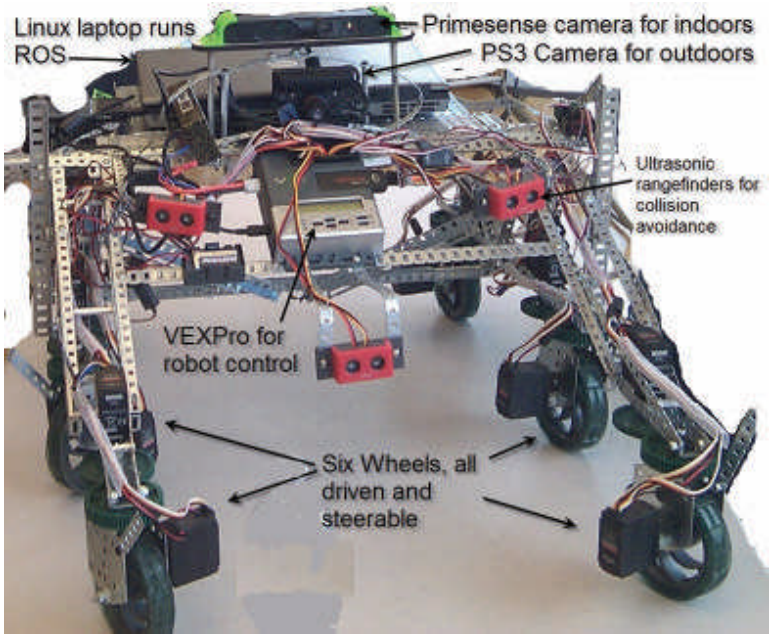


FIGURE 1. jRover (labled).

The sexiest robots today are the Mars rovers — Curiosity and Spirit — and the Google self-driving car. We thought, "Why not mash-up those ideas and see if we could build a robot that could navigate the Roborama course?" Our robot needed to be large enough to stably carry a laptop while traversing rocky ground, yet small enough for a person to pick up and carry. VEX Robotics' line of metal and motors were scaled about right for the job. If we added lane-following technology from self-driving cars to keep the robot centered between the course markers, we thought we might have a competitive entry.

You can't just buy a lane-tracking vision system off the shelf. However, you can get ROS — the Robot Operating System — which includes *opencv* and other vision packages. We were able to easily build a lane-tracking vision system using freely available packages. (You can too if you use the packages and source code provided.)

System Design

Flexibility was a key design goal. We wanted our robot — named jRover — to be a long-lived platform for indoor and outdoor missions. We decided jRover should be self-contained — it should not rely on an external computer. That meant it had to carry a laptop to run Linux and ROS. Using a camera and vision software as the primary navigation subsystem maximized responsiveness to the unknowns in the environment.

We also wanted several fallback strategies for the contest in case the lane-following idea didn't work. These included having the robot use ultrasonics or vision to follow someone who walked the course. Accordingly, we equipped it with a range of sensors.

The motors and ultrasonic ranging sensors were wired to a VEXPro robot controller which communicated with the

ROS laptop over Wi-Fi. The PS3-eye camera and Primesense RGBD camera were cabled to the laptop.

We chose to use ROS mainly because of the vision processing packages available, but it brought additional benefits such as easy communication between different software packages, sophisticated logging, and the ability to replay video from a run to debug software.



FIGURE 2. Paul and Jason debate jRover chassis design.

Mechanical and Electrical Design

We were inspired by the Spirit rover which (at the time) was about to land on Mars. As a showpiece at DPRG outreach events, it would be instantly recognizable to children and adults alike. We also thought the rover mechanical design would provide great handling on rough terrain. It used six-wheel drive to maximize drive power from small light motors. All six wheels were steered which enabled Ackerman-style cornering, as well as crabbing. To steer, we turned the front and back wheels in opposite directions, which afforded a near zero radius turning ability. We built several iterations of the robot as we found and addressed various challenges before arriving at the final design.

The first challenge was the suspension. So that the Mars rover can climb over rocks as big as twice its wheels' diameter — all the while keeping six wheels on the ground — its rocker-bogies have stub axles for each wheel and an independent suspension that drives a self-leveling platform. We were unsuccessful in our attempt to duplicate that aspect of the design; the weight of the laptop on the platform stripped the plastic gears that were supposed to keep the platform level. We ended up tying both front wheels and the platform together into a single rigid assembly which resulted in sometimes only having five wheels on the ground.

Another challenge was the weight of the laptop which caused flexing in the lower chassis structures. That problem was compounded by the wheels which had straight sides and didn't slip sideways on grass or carpet (as they must to avoid deforming the chassis). Also, they were steered with motors which had considerable backlash compared to servos. We had decided to use motors for steering because they're stronger, but the backlash meant the wheels were never perfectly aligned.

Adding to that problem, the integrated motor encoders made backlash-induced misalignment undetectable. Also, the wheels were mounted on beams extending from the bogie pivot point, and the beams would twist. The result of all this was the wheels would run in or out from their intended track, twisting the beams as they did. We never solved these problems satisfactorily.

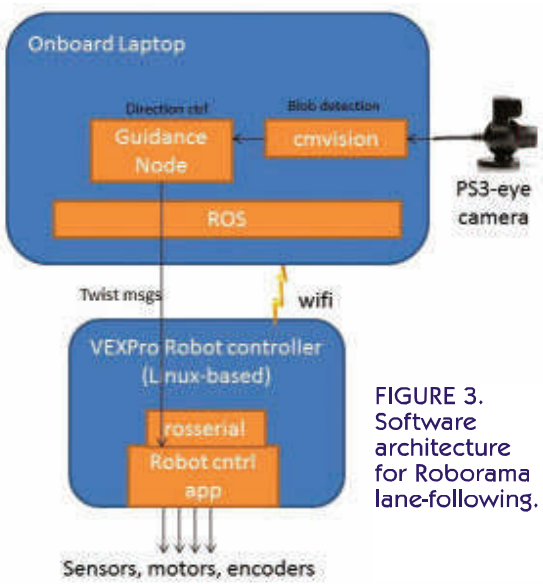


FIGURE 3. Software architecture for Roborama lane-following.

Motor speed and steering motor position were determined from encoders integrated into the VEX motors. Determining "straight" for the wheel-steering assembly was a challenge since there was no indication of absolute position. We installed a mechanical stop on the steering head, and at startup, we drove the steering head against the stop, then rotated the head back a fixed amount that was calculated to center it.

We also knew that when turning we had to make each

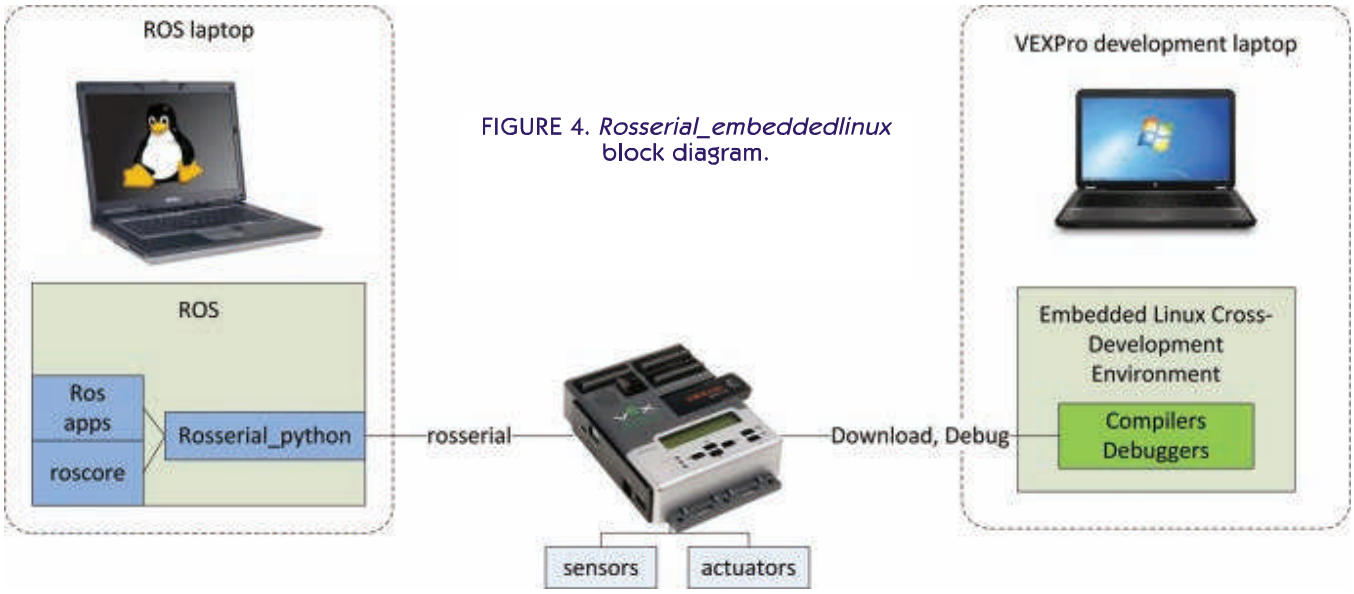


FIGURE 4. *Rosserial_embeddedlinux* block diagram.

wheel run at a different speed — depending on how far it was from the vehicle's center of rotation — in order to maintain traction between the wheel and the ground. We used the drive motor encoders for closed-loop speed control on each wheel individually, calculating each wheel's required speed based on turn radius.

In addition to the PS3-eye camera and motors with integrated encoders, we installed sensors intended for future use, such as an android phone providing access to GPS and a gyro, and an Xtion Pro depth-sensing camera for indoor VSLAM use.

Software Design

Figure 3 shows the overall organization of the software components. The vision software used *CMVision* — Carnegie-Mellon's blob-tracking package which integrates with ROS to publish a list of blobs and their size and location. Documentation is at **www.ros.org/wiki/cmvision**. *CMVision* can be installed with *apt-get* — like any other ROS package. We mounted a camera on the robot and used *CMVision* to detect the course boundary flags. *CMVision* comes with a great configuration GUI that selects a range of colors to define a "blob of interest."

This was very useful because the camera reported different colors for the flags, depending on lighting conditions — indoor, outdoor cloudy, sunny, dusk, etc. After configuration, *CMVision* started publishing blob data corresponding to the boundary flags at 10 Hz. Blob tracking with no work! We wrote a simple ROS Python guidance node which exerted closed-loop control over robot direction. It subscribed to the blobs topic published by *CMVision*, and looked for the furthest left and right blobs. The guidance node controlled robot drive direction so as to guide the robot to the midpoint between the outer blobs — assuming they would be the nearest track markers seen by the camera.

The guidance node drove the robot by publishing twist messages — a standard ROS message type — which the robot control app running on the VEXPro controller subscribed to. In under 400 lines of Python code, we had closed-loop control based on vision input! You can download the code from the DPRG ROS source code repository at **https://code.google.com/p/dprg-ros-pkg**. The jRoverRos directory contains the *CMVision* configuration files, and jRoverRos/bin contains the guidance node code.

The VEXPro had connectors for electrical interfacing to VEX motors and sensors, and brought the power and flexibility of its Linux software environment. It had good networked debugging support, and was able to run multiple processes. However, it did not have enough horsepower to run the full ROS environment. We ported the *rosserial_arduino* package to Linux, which enabled VEXPro software to interact with ROS nodes running on the laptop. The laptop ran ROS, read camera data, did the vision processing, and ran the guidance node, while the VEXPro ran the low-level motor and encoder software.

You can find links to documentation and tutorials for *rosserial* for Arduino or embedded Linux at **http://ros.org/wiki/rosserial**. If your robot controller uses Arduino or embedded Linux, it is straightforward to connect it to ROS, and for it to receive speed and direction messages from the

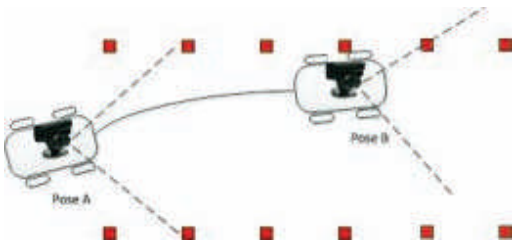


FIGURE 5. Effect of camera/robot misalignment.

Video with more details of the mechanics and system components, and the robot in early operational testing can be seen at **www.youtube.com/watch?v=QT-CTaU7O8E**.

guidance node. **Figure 4** shows how we connected the VEXPro to ROS.

Rosserial_python connects via serial or Wi-Fi to the VEXPro robot controller. You have source-level debugging under the Integrated Development Environment (IDE) on Windows for quick debug and single-stepping through robot code running on the VEXPro. By far, the largest and most challenging part of writing the software was robot control on the VEXPro — controlling the numerous motors and encoders. You can find the source in the *jrover/vexpro* directory of the DPRG ROS source code repository; however, your robot is almost certainly different. If you have software to run your robot, you need only to subscribe to Twist messages using *rosserial* or otherwise. Use the Twist messages to control your robot drive motors if you want to duplicate just the lane-following part of this design.



FIGURE 6. Contest day in the robot pit, on the shores of Lake Lewisville near Dallas, TX.

System Testing

We set up a test course in a nearby park, and set jRover on its way down the track. On sunny days, we had good success getting *CMVision* to see the flags. Then, the weather turned overcast, and the inexpensive webcams we were using completely failed — every model we tried whited out. There was insufficient contrast in anything we pointed them at. Finally, in desperation, we tried a Sony PS3-eye camera and it worked beautifully. The better camera provided clear images at dusk, as well as in full sunlight that saturated the less expensive cameras. We learned to use a good camera.

We struggled to get the robot to stay in the center of the course. We adjusted the gain of the control loop to no avail — the robot would veer to the same side of the course, and eventually run into one of the course-edge markers or turn completely out of the course. After much experimentation, we discovered the importance of bore-sighting the camera — meaning, to align the direction the camera is pointing with the direction the robot will run when it's told to drive straight. We hadn't understood how the system would work when the camera center was not aligned with robot "straight." **Figure 5** shows what happened in this situation.

When the robot was in pose A, the camera saw two flags and the guidance node commanded the robot to go straight. It headed toward the side of the course because it was misaligned. If the robot should happen to straighten up, the camera would have turned with the robot and would be looking across the course; the guidance node would command the robot to go left, driving it further toward the edge of the course. The error grew without bound, and the closed-loop control wouldn't correct it.

You can watch video of the robot driving down the Roborama course at **www.youtube.com/watch?v=xKXq-ZeOCnc**.

In pose B, the camera still saw two flags, but it was now looking at an angle across the course. The guidance software again said go straight, but the robot was very near the edge of the course and would easily wander out of bounds as the right flag slipped out of its field of view and the presumed center of the course moved to the left. After we aligned the camera with robot "straight," it would drive down the center of the course.

Contest day was upon us by the time we made the lane-following logic work on the straight-away. Next, we needed to make it turn the first corner. We adapted the corner by wrapping it in red tape and distinguished it by blob size. This enabled jRover to make it around the first corner, but the vision system was then fooled by the large area of flags at the hairpin bend that followed.

In the final analysis, our attempt to mash-up a Mars rover with the self-driving car was partly successful, but showed us plenty of opportunities to improve. Our mechanical design needs to be made stronger and accurate, and our vision system design is too simplistic. However, we learned a lot and laid a strong foundation on which to build improvements for the next contest. SV

AP CIRCUITS
PCB Fabrication Since 1984

As low as...
$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com

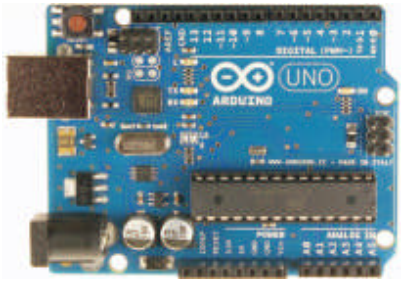
VISA MasterCard PayPal IPC MEMBER BBB

CanaKit™
Electronic Kits • Modules • Parts

Save 10% using Coupon Code SM2012* when ordering online at www.canakit.com

Motor Controllers • Arduino • Arduino Kits • Arduino Shields • Robot Platforms • Parts

Arduino Uno

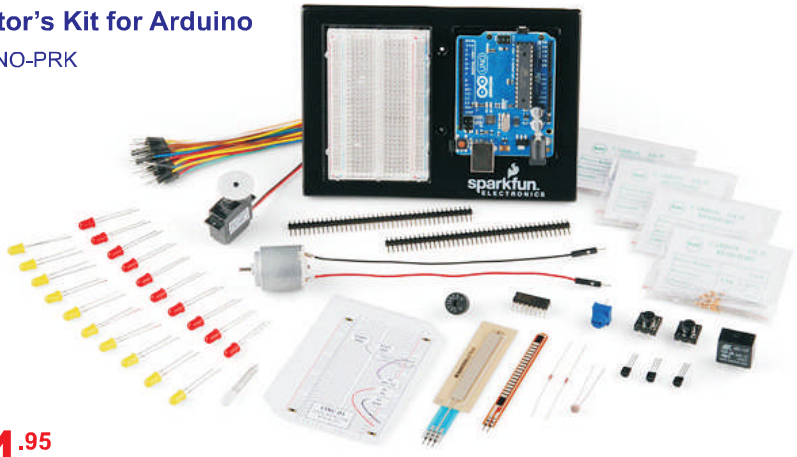


$29.95 ARDUINO-UNO

Latest "R3" Version!

Inventor's Kit for Arduino

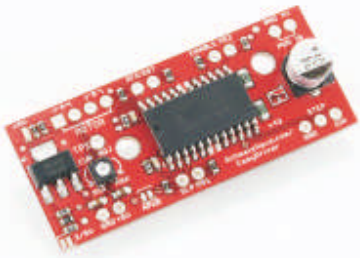
ARDUINO-PRK



$94.95

Includes 36-Page Printed Full-Color Manual

EasyDriver Stepper Motor Driver



SX09402 **$14.95**

Stepper Motor



SX09238 **$19.95**

50A Motor Speed Controller (PWM)

UK1133



Optional LCD



$59.95

7A Bi-Directional Motor Speed Controller (PWM)



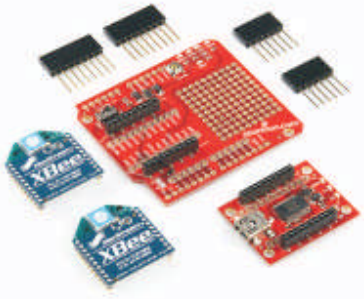
UK1125 **$39.95**

USB PIC Programmer (MPLAB Compatible)



UK1300 **$69.95**

Xbee Wireless Kit



XBEE-KIT **$94.95**

1•888•540-KITS (5487) info@canakit.com

www.canakit.com

* Limit one per customer. Can not be combined with other coupons or volume discounts. Does not apply to products already on sale. Prices subject to change without notice.

3D Printers

This month:

Part 1. Introduction into 3D Printers

Part 2. Assembly Highlights

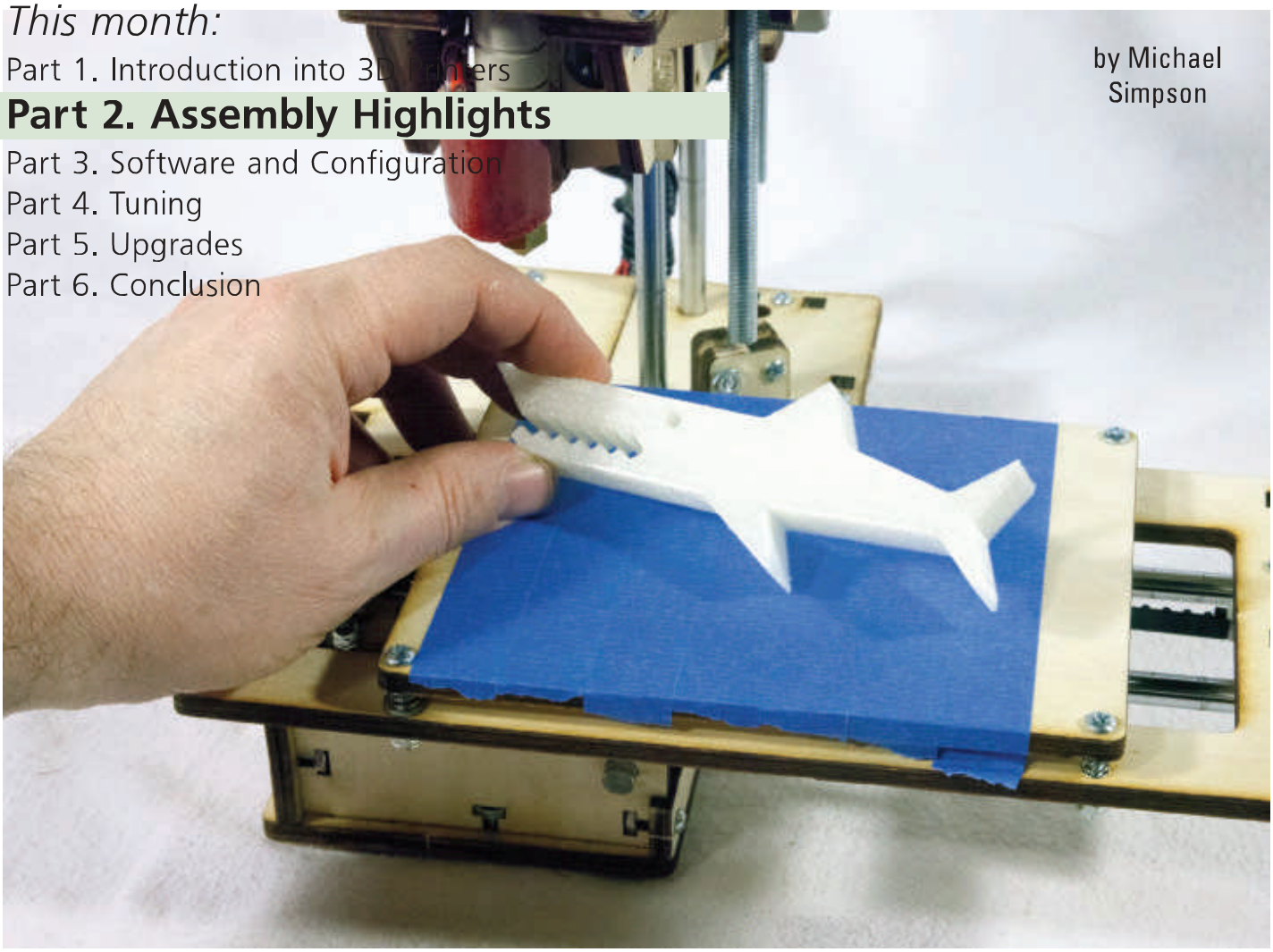
Part 3. Software and Configuration

Part 4. Tuning

Part 5. Upgrades

Part 6. Conclusion

by Michael Simpson



www.servomagazine.com/index.php?/magazine/article/june2013_Simpson

Discuss this article in the *SERVO Magazine* forums at http://forum.servomagazine.com

Last month, I explained a little about the 3D printing process and the parts that make up an extrusion type printer. I then briefly introduced you to three 3D printers: the Printrbot Jr., Rostock Max, and Solidoodle 2.

This month, I am adding a fourth printer to the mix. The printer shown in *Figure 1* is an Afinia H-Series printer. It's fully assembled and ready to print out of the box.

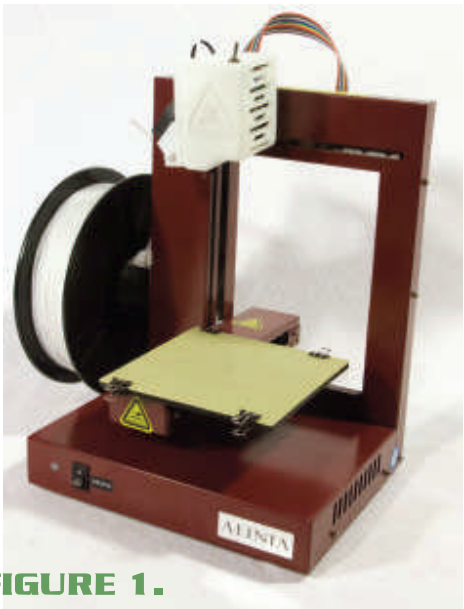


FIGURE 1.



FIGURE 2.

In addition, the printer may only be available in kit form. The Rostock Max falls into this category.

I used to build model airplanes back in the day. I can remember hours of enjoyment constructing each craft, and building 3D printers can be just as enjoyable if you take your time and follow the instructions. Taking time to assemble things right will reward you with a 3D printer that creates better "prints." It isn't a race, so take your time. While assembling your printer, if you encounter a problem don't be afraid to ask questions on the manufacturer's forums. That's why they are there.

Assembly Highlights

Let's take a look at some of the pictures I took as I assembled the Printrbot Jr. and the Rostock Max 3D printers. There are three reasons why you may want to assemble a 3D printer from a kit:

1. *Savings.* On average, you can save up to $100 by assembling the printer yourself.
2. *Fun.* You enjoy building things.
3. *Knowledge.* You want to learn and understand the interworkings of your 3D printer.

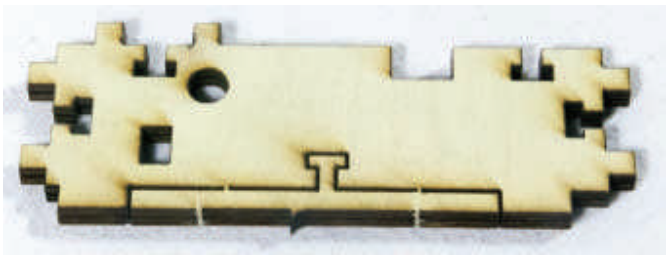


FIGURE 3.

PrintrBot Jr.

I ordered the PrintrBot Jr. this January 28th and it arrived 12 days later. The parts (shown in **Figure 2**) were well packed.

The PrintrBot Jr. is normally shipped with a laptop AC adapter, but they were out of stock and sent me an ATX power supply which is a required upgrade if you want to add a heated bed to your printer. They later shipped me the laptop AC adapter when it was in stock. In addition, the main extruder gear was warped so they had to send me a replacement, along with some missing lock nuts.

The printer frame is made from 1/4" five-ply birch plywood as shown in **Figure 3**. The parts are laser-cut and fairly easy to remove from the sheets. They contain small tabs to hold them in place, so some cleanup of the parts will be needed.

You will need only basic tools like the ones shown in **Figure 4**. While you can get by with a Phillips screw driver, a power drill will make the assembly go faster. The file and razor knife are used for part cleanup.

You can find instructions for building the Printrbot Jr. at **http://printrbot.com/support/instructions-guides**. You can also view the online document as a PDF. This will allow you to download it to your computer.



FIGURE 4.

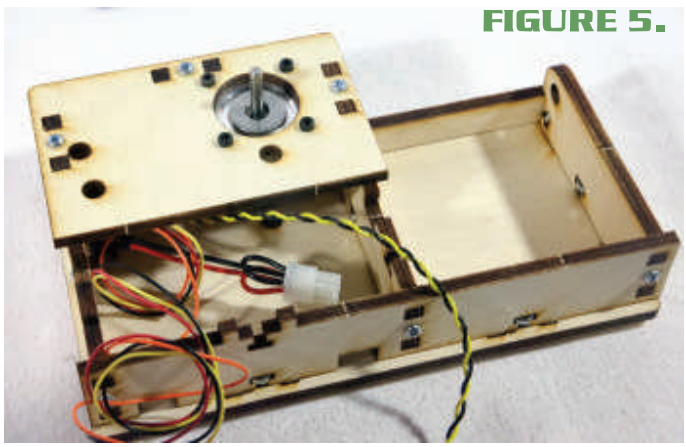


FIGURE 5.

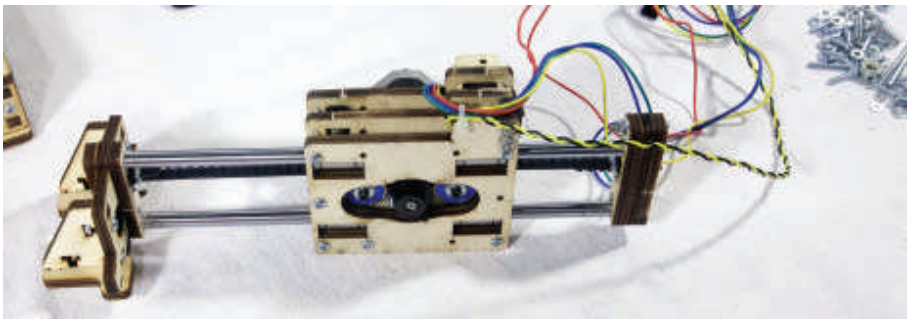


FIGURE 7.

The instructions state that you will need 1-2 hours to complete the build. I suspect this estimate was from someone who has built the machine several times. My estimate is more in the order of 4-12 hours.

The instructions will take you through the assembly of the base (**Figure 5**), table platform (**Figure 6**), and the main arm (**Figure 7**).

The extruder shown in **Figure 8** is used on several of the Printrbot 3D printers. For this reason, it has its own assembly instructions.

The controller board is installed next. You will connect the various connectors to the board (**Figure 9**), and then install the board into the base as shown in **Figure 10**.

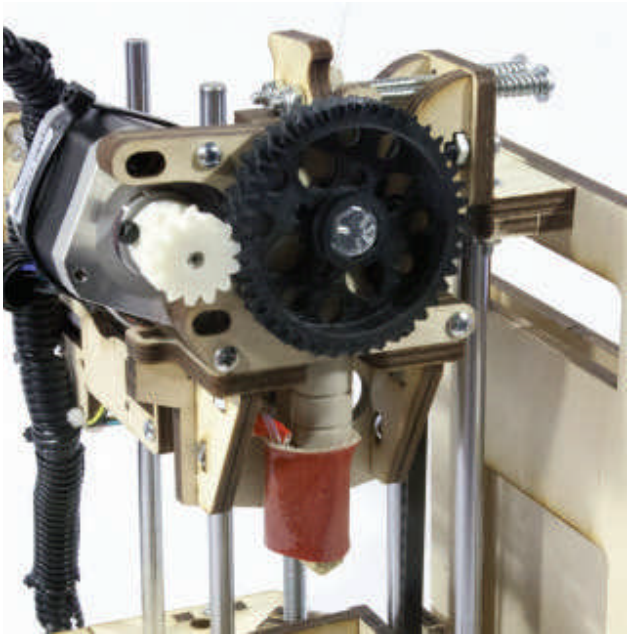


FIGURE 8.

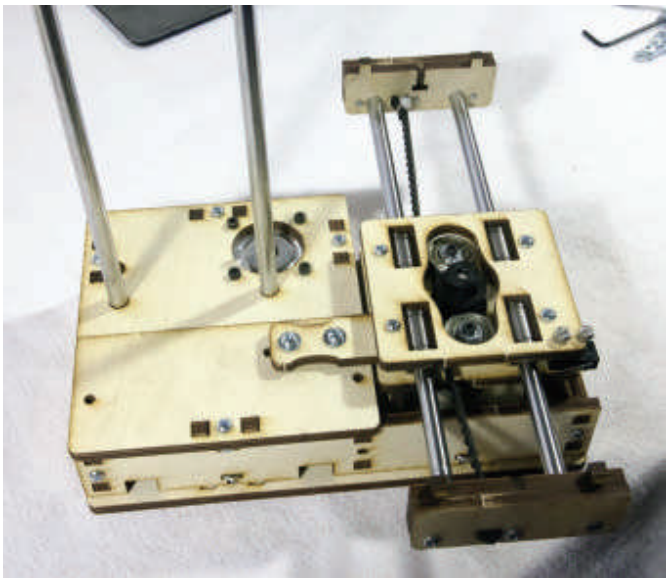


FIGURE 6.

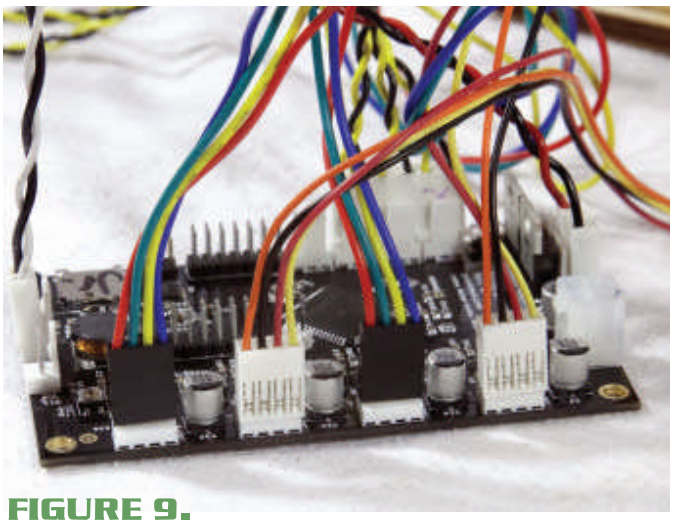


FIGURE 9.

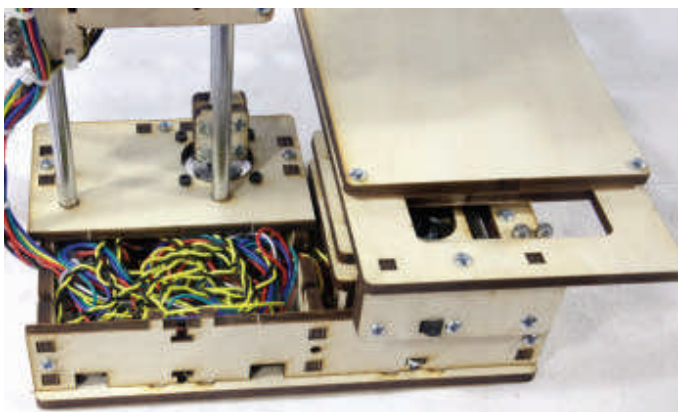


FIGURE 10.

The addition of the controller board will conclude the assembly of the Printrbot Jr., but you still need to install the software and level the table. This is covered in the *Getting Started Guide* on the Printrbot website. As a final touch, I recommend the use of split loom cable to cover the wires as shown in **Figure 11**.

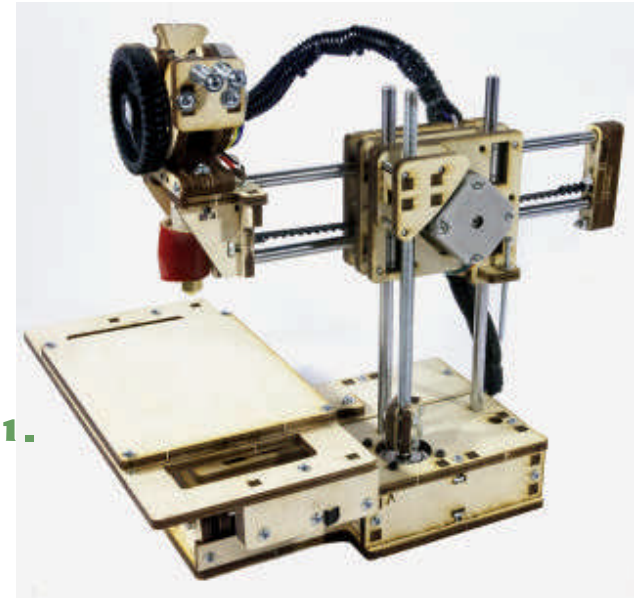


FIGURE 11.

Rostock Max

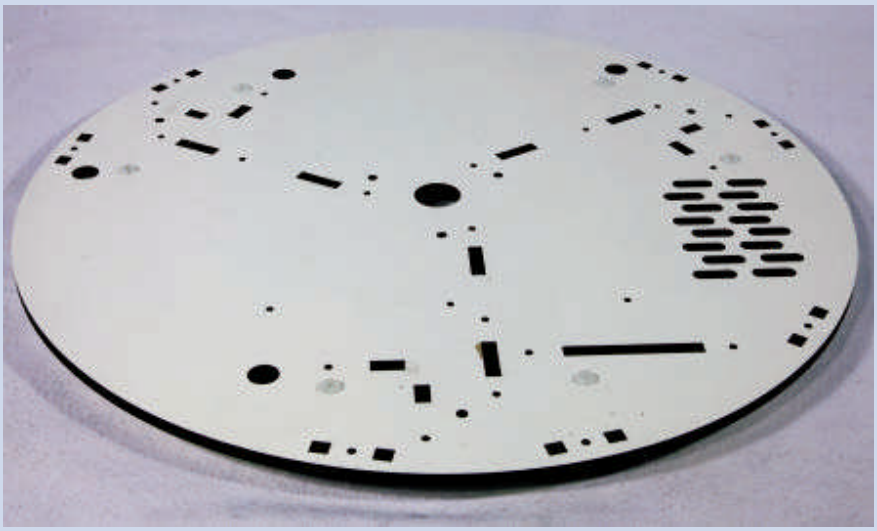
I ordered the Rostock Max this February 4th. It arrived 11 days later. The parts (shown in **Figure 12**) were also packed very well and complete. Everything was clearly marked and easy to identify.

The printer frame is made from 1/4" melamine coated MDF shown in **Figure 13**. The parts are laser-cut and fairly easy to remove from the sheets. The parts come held together with masking tape, and are wrapped in a protective covering. Both the masking tape and protective covering must be removed from the parts. This process took me over an hour to accomplish.

The tools required to build the Rostock are similar to the Printrbot Jr., with the exception of a couple of extra wrenches needed for some of the bolts.



FIGURE 12.



You can find instructions for building the Rostock Max at **https://github.com/geneb/Rostock-MAX-Docs**.

The assembly manual is put together very well, but lacks step-by-step instructions for assembling the extruder. Instead, they point you to a very low resolution video. This was the most confusing aspect of assembling the Rostock Max, but you should be able to work through it. The Rostock Max will require about 24 assembly hours.

FIGURE 13.

The instructions will take you through the assembly of the base (**Figure 14**), belt guides (**Figure 15**), platform and power supply (**Figure 16**), and uprights (**Figure 17**).

Next, you will install the Rambo controller board and wire the motors, homing switches, and heated bed (**Figure 18**). You will need to assemble the extruder and install it on the Rostock (**Figure 19**); then comes the three carriages. They need to be assembled and installed as shown in **Figure 20**.

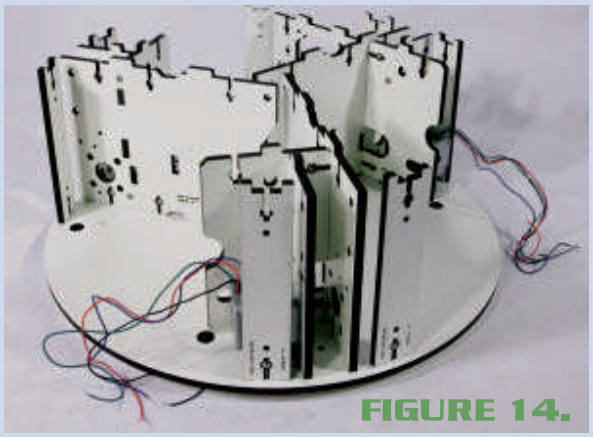


FIGURE 14.

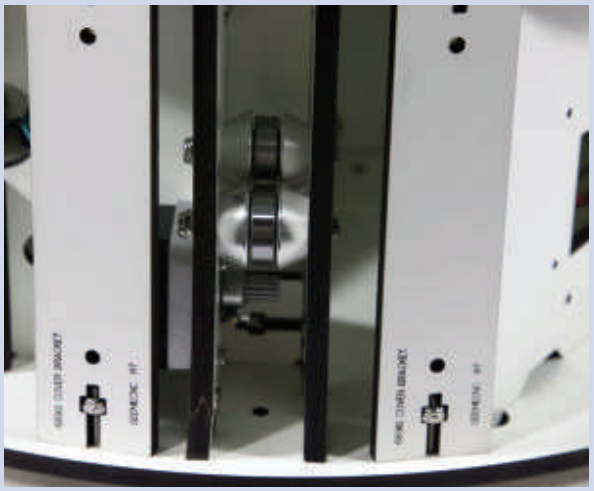


FIGURE 15.



FIGURE 17.

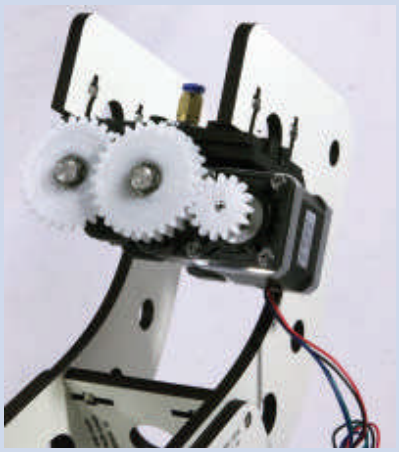


FIGURE 19.



FIGURE 16.



FIGURE 20.

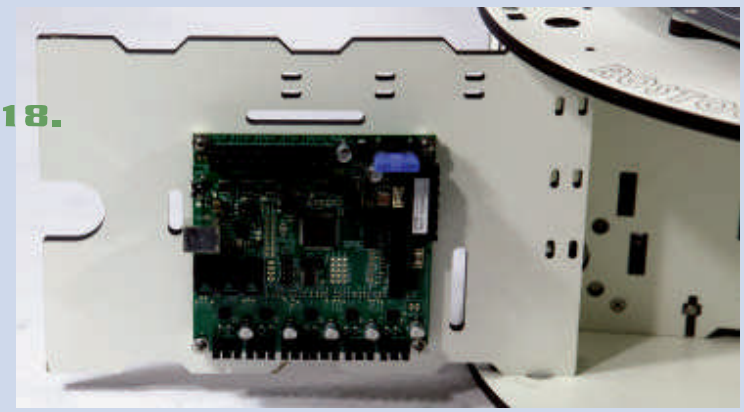


FIGURE 18.



FIGURE 21.

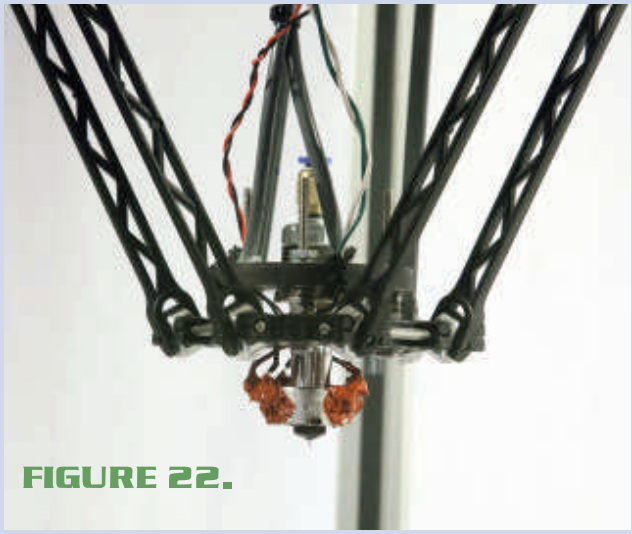


FIGURE 22.

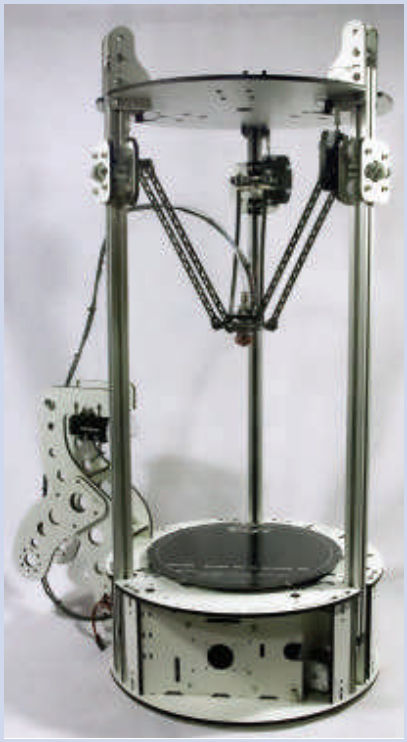


FIGURE 23.

The delta arms and platform shown in **Figure 21** are next. This is a time-consuming process, as you need to use some sand paper to sand the ends of the arms until you got a smooth — but snug — fit. It's a sand-test-repeat process. Once you are satisfied with the results, the other ends of the arms are attached to the side carriages.

The last step is to assemble and install the hot end shown in **Figure 22**. The hot end assembly requires RTV silicone, Kapton tape, and aluminum foil. These are not included with the kit, so you will have to purchase them separately. The RTV must be a high temperature type, and can be purchased at any local auto parts store. The Kapton tape, however, will need to be purchased online.

The completed Rostock Max will look like the printer in **Figure 23**. As before, you still need to install the software and level the table. Leveling the table for a delta robot is time-consuming; but the assembly manual will take you through the process. The manual also includes a step-by-step guide for setting up the software.

Printer Unboxing

Solidoodle 2

I ordered the Solidoodle 2 March 3rd and received it five days later. The Solidoodle 2 is shipped with foam padding in a very roomy box as shown in **Figure 24**.

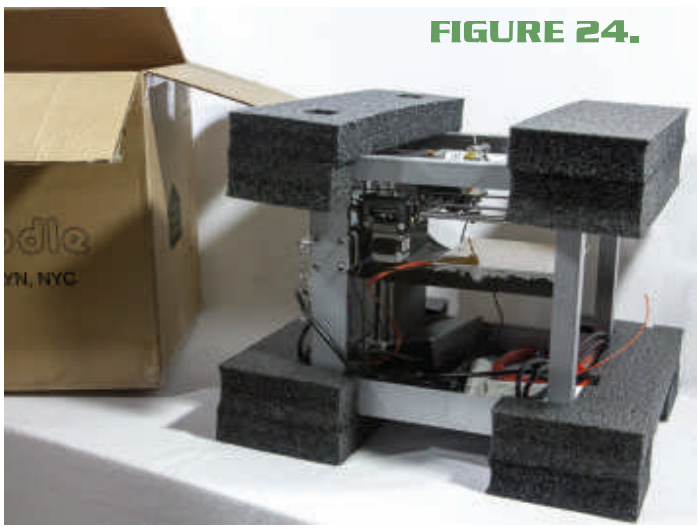


FIGURE 24.

The box contained the following items (**Figure 25**):

- Solidoodle 2
- AC adapter
- USB cable
- .3 ounces of filament (very small amount)
- Filament holder parts

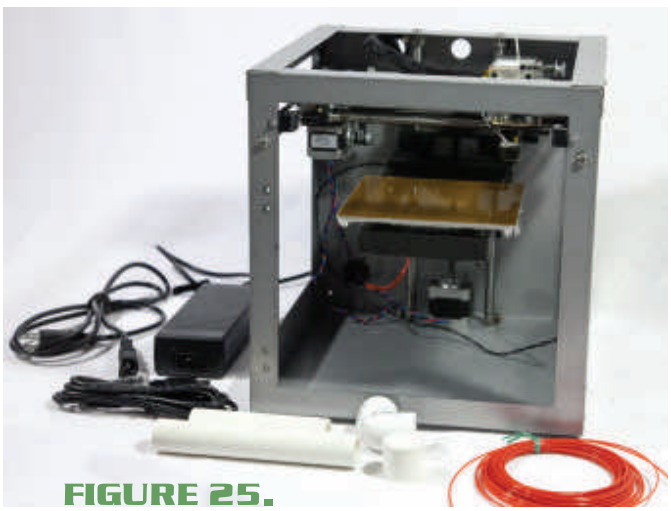


FIGURE 25.

You will need to assemble and install the filament holder onto the back of the Solidoodle 2 as shown in **Figure 26**. The parts are pressed into place, so there is no need to glue them. I purchased the Solidoodle 2 Pro shown in **Figure 27**. It comes with a heated bed, improved filament mount, upgraded power supply (needed for the heated bed), and interior lighting.

Afinia H-Series

The Afinia H-Series printer that I have is a loaner, so I can't comment on the lead time needed to purchase one of these. I will say the unit I received was new, so the unboxing is the same. What's in the box?

- Afinia H-Series 3D printer
- 700 g of premium filament
- Work gloves
- Instruction manual
- Software CD
- USB cable
- AC adapter
- Scraper
- Razor knife
- Cutters
- Parts for filament holder
- Three perforated build boards
- Three Allen wrenches
- Extruder cover
- Filament tube



FIGURE 28.



FIGURE 26.

The 3D printer shown in **Figure 28** does not have the filament holder installed. Using one of the provided Allen wrenches, you will need to remove two screws on the left side of the Afinia. Next, you will attach the two included plastic parts using the removed screws. The plastic parts hold the filament roll in place and provide a mechanism for feeding the filament to the extruder through a thin tube.

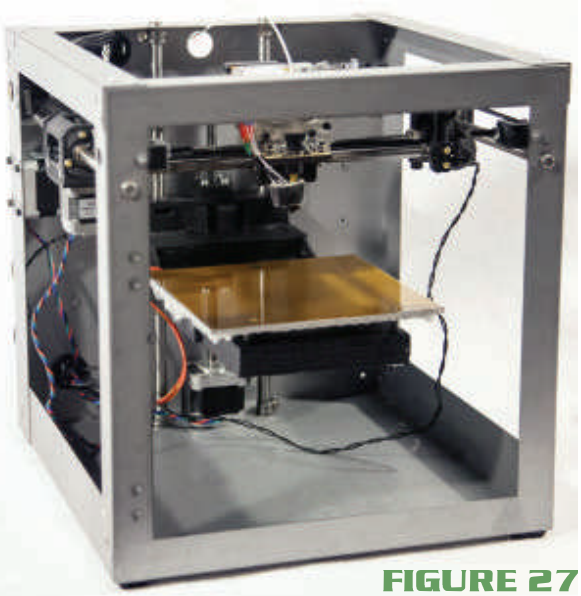


FIGURE 27.

Conclusion

At this point, I have all four printers assembled. The next step is to install the software and attempt our first print. Next month, we will do just that. I will take a simple object and print it on all four printers.

Before closing, I want to give you a cost breakdown for the printers mentioned:

- Printrbot Jr. Kit $399
- PrintrBot Jr. Assembled $499
- Rostock Max Kit $999
- Solidoodle 2 $499
- Solidoodle 2 Pro $599
- Afinia H-Series $1,599

Please note that these do not include the shipping charges, and can be subject to change. Please be sure to post any questions in the *SERVO Magazine* forums at **http://forum.nutsvolts.com/viewtopic.php?f=49&t=16968**. I will also be posting additional information on my website at **www.kronosrobotics.com/3dprinters**. SV

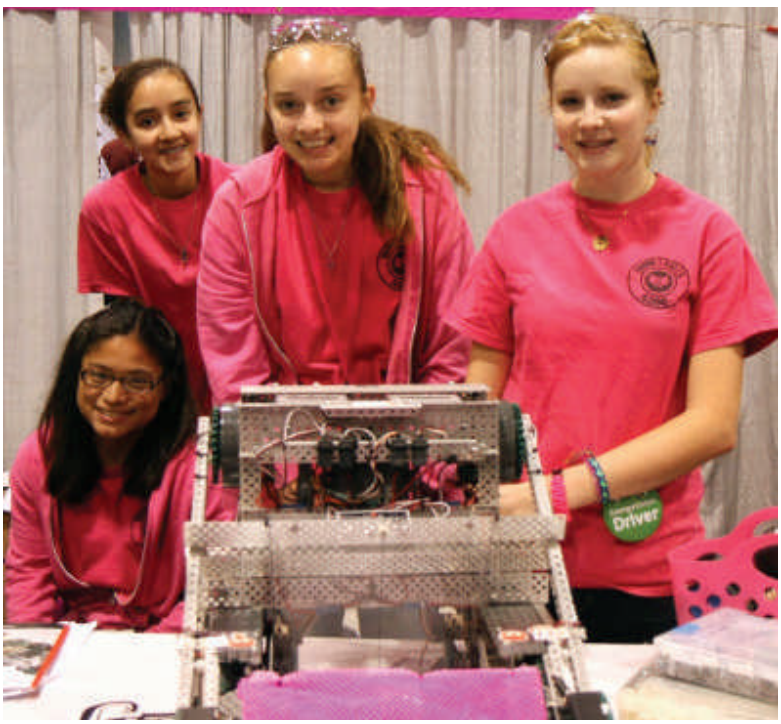
The VEX Robotics 2013 World Championship



More than 700 teams from 24 countries gathered at the Anaheim, CA Convention Center for this year's VEX Robotics World Championship. With over 15,000 student roboticists present, it was a very busy three days.

In addition to the high school and university level events, VEX introduced a pilot program for elementary school teams. The new program will be called VEX IQ. This was my fifth year photographing the VEX Championship and these are just a few of the hundreds of photos I shot. You can see the rest on flickr at **http://goo.gl/LRhgc**.

— R. Steven Rainwater



The Omni Eagles team.



Al Bayan Model School team 7321.



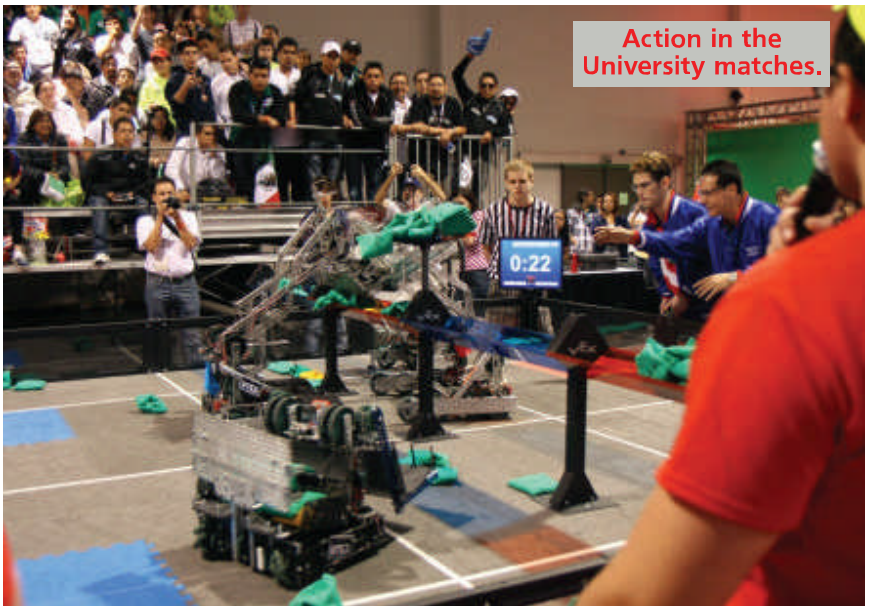
Disco girl from team Discobots.

VEX mohawk.



Kohala Robotics team in final matches.





Action in the University matches.



Viking headgear.

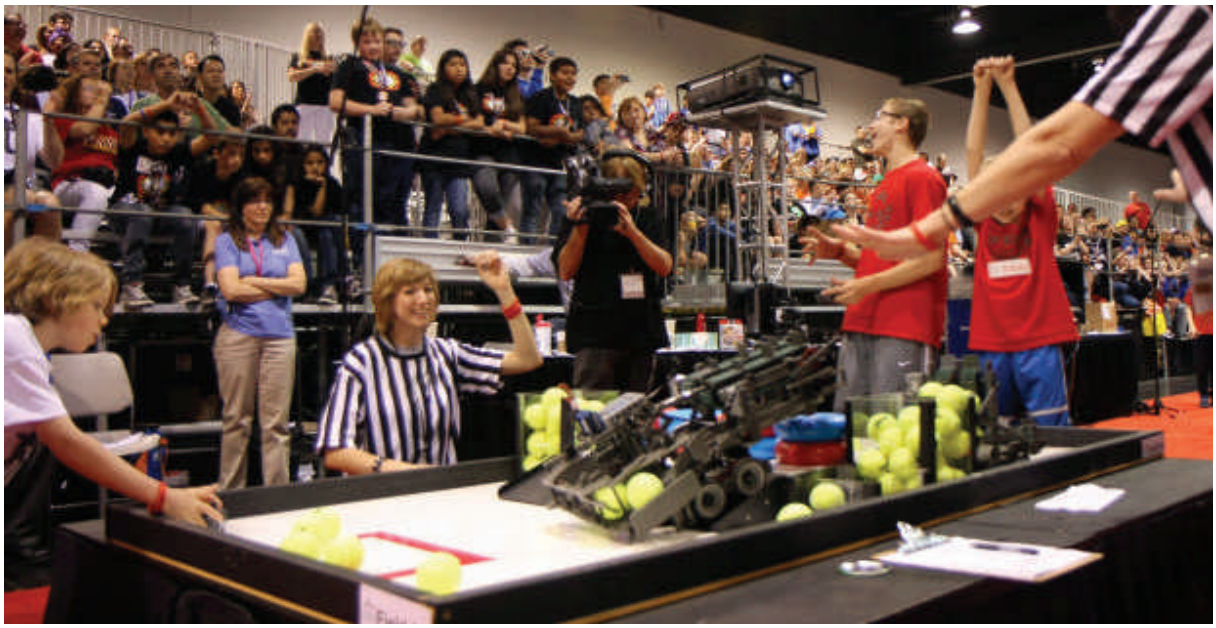


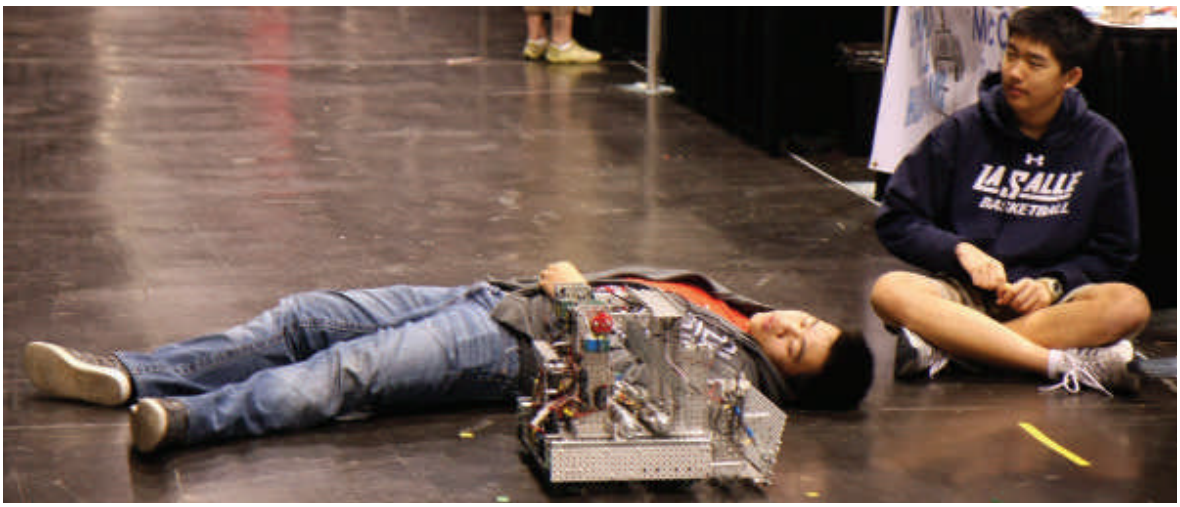
Parade of teams during the opening ceremonies.

Small portion of the team pit area.



Action in the Pilot Program for younger kids.





Exhausted team members taking a break between matches.

Team watching their robot in practice area.



Team members having fun between matches.





Then and NOW

Robots and the Law

by Tom Carroll

Back in April of 1984, I was one of a dozen or so speakers at the International Personal Robot Congress in Albuquerque, NM. Isaac Asimov delivered the keynote address via a video TV link from New York — much to the disappointment of those who had hoped to meet the famous author in person. Joseph Engelberger was another key speaker. My topic was about robot hardware, and I believe most of the attendees managed to stay awake during my presentation.

Towards the end of the three-day conference and exposition, all of the personal robots from exhibitors and hobbyists were lined up outside the front of the exhibition hall for a group photo. As I was standing behind the photographer, an elderly woman came up to me and said, rather angrily, "What are you all doing here, trying to make mechanical men? Only God can make people. You should be ashamed of yourselves. Do you plan to have these things take over the world?" I was a bit taken back by her statements. This was in the days before walking bipedal humanoids that we have today, so none of the robots really resembled a human. I doubt that she was a participant in the conference or had actually been inside to view the exhibits. Her 'agenda' to scold all of us left me unsettled. Her precise wording escapes me — as does my actual response to her — but the feeling of her admonishment to me remains vivid to this day.

Just what did she and the rest of the world think that robot builders were trying to accomplish? Did she feel that we were intent on building robots to overthrow Mankind? The movies of the *Terminator* series and *I, Robot* were many years in the future.

Suddenly, I felt that all of the attendees had somehow been transformed from people who enjoyed studying and building robots into a group of mad scientists intent on destroying the Earth with our evil creations.

These comments were made almost 30 years ago, and you could certainly sense the anger and distrust that this woman expressed about robots. Here we are in the 21st century and one would think that with many decades of exposure to robots, the public would be used to them in all aspects of our life. Not so.

Marvin Rosenblum, an attorney who specialized in high tech start-up companies back in the early '80s was a speaker at the 1984 IPRC. Even in those early days of robot use, the legal implications of robots were clearly in the minds of early developers.

The use of robots has created so many legal issues these days that major robot manufacturers must employ a legal team to carefully examine the use and subsequent liabilities of a company's robotic products.

Robots and the Law

Last September, Tom Green, Editor in Chief of *Robotics Business Review*, and author Emmet Cole wrote an in-depth article entitled *Robots and the Law*. Tom began the article with these words: "**Humankind's new tool: Who gets the blame when one screws up?"**

Another quote that Tom made from the film, *2001: a Space Odyssey* —*"Open the pod bay door, HAL." "I'm sorry Dave, I'm afraid I can't do that"* — has been the brunt of many a joke since the movie debuted in 1968.

Figure 1 shows one of hundreds

of photographs and cartoons that have poked fun at that critical scene in the movie when the onboard computer (HAL) malfunctioned and would not allow the remaining live crewmember (Dave) back into the spacecraft.

Could HAL have been taken to court and be convicted of killing four people and trying to kill a fifth? The way the movie ended, who knows. However, it makes one wonder about the many legal consequences of the use of robots.

"Humans are tool builders," wrote Green. "We're seemingly hard-wired to fashion objects that do jobs for us or make life better or easier, or both. Some of these tools — the ones we call machines — have been absolutely necessary in building and maintaining our civilizations, our modern way of life, and the ability to plan for our future."

"Our relationship with our machines — however beneficial and salubrious — is many times an uneasy and often contentious one, especially so since the Industrial Revolution. Railroads are wonderful until a locomotive slips its track. After the wreckage is hauled away — the injured and dead, as well — another of humankind's great inventions takes over: Law."

"The rules we all use to interact with each other and the system by which we can seek to be made whole when we are wronged, intervene to try to understand what took place and to assess liability, arrive at a judgment, and then determine punishment and reward, if any."

In my opinion, Tom and Emmet stated the legal side of new technology extremely well in these opening sentences of their article.

Robotics Industries that Could be at Risk

The media has been awash with news about the public's distrust of robots. Intuitive Surgical — makers of the popular da Vinci surgical robots — is facing a series of serious lawsuits concerning ostensible problems with their products. Private citizens are angry at local, regional, and our national government about the dangers (as well as the potential personal rights and privacy invasions) that may be created by drones.



FIGURE 1. "Open the pod bay door," from *2001: a Space Odyssey*.



FIGURE 2. Sonny (from the film, *I, Robot*) being interrogated in a police station.

Autonomous, self-driving robots are welcomed by some and feared by others as being potentially lethal on our country's highways. Of course, lawyers are waiting in the wings for the many suits that they assume (hope!) will be filed against robot manufacturers. I will address these legal implications a bit later.

Unmanned Aerial Vehicles

I wrote a bit about UAVs in last month's column, and showed an image of (supposedly) anti-drone/stealth wear that can hide us from those ever-seeing evil robots above us. I've heard people moan, "These new fangled flying killer spies will bring us down to the police state level of North Korea." It seems that new technology always has its detractors. When horseless carriages first appeared, people were appalled at the noisy things that smelled of gas, scared everyone's horses, and put many animal feed suppliers out of business. Even today's movies play up mistrust in robots, such as with Sonny in *I, Robot*. **Figure 2** is a scene from the film when Sonny was being interrogated as a murder suspect and kept insisting he was innocent (even though, in actuality, he wasn't).

There are independent groups that enjoy stirring up malcontent about any new technology, and robots seem to be in their crosshairs now. Military versions are viewed as autonomous killers preying upon unsuspecting citizens, whereas the civilian drones are considered airborne government spies.

I've heard people say things like "It's bad enough these things are above us, spying on our every activity, but now they can carry missiles so the government can take out any person they choose — whether guilty or innocent." I feel comments such as these serve no useful purpose other than to stir up discontent. However, I believe we need more trustworthy people in today's media, not just to report the truth about the robotics field, but for all aspects of life.

Robots Above Us

Drones have been around and in the news for years now. It seems that very few of the positive features and uses of these UAVs have been praised. Police departments — large and small — have begun to use smaller quadcopters with a video camera to perform surveillance on crime, accident, and fire scenes.

Figure 3 shows a typical police model made by Nord Atlantic in Miami, FL. Certainly, these types of aerial vehicles can be used to spy on



FIGURE 3. Nord Atlantic Police UAV.



FIGURE 4. Boeing Phantom Eye UAV.

unsuspecting people in an unauthorized manner, but reliable reports reveal that these illegal occurrences are few in number.

Aerial surveillance is extremely valuable in military and battle conditions, and not all military UAVs carry weapons. Boeing's unmanned Phantom surveillance drone shown in **Figure 4** is capable of flying over battlefields for four days straight. Powered by two highly modified Ford Fusion 2.3 liter diesel engines modified to run on hydrogen, the 150 foot wingspan plane can carry 1,900 pounds of liquid hydrogen fuel and over 500 pounds of equipment. By flying as high as 65,000 feet using superchargers to compress the thin air at altitude, the UAV can be 'on station' 24/7 to assist battle theater commanders.

Autonomous Automobiles Come of Age

Let us leave the contentious subject of drones behind us, and look at an emerging technology that will soon bring hungry lawyers out in force: self-driving automobiles. Two recent articles — one in *Wired Magazine* entitled "Leave the Driving to Us" by Clive Thompson, and the other in *Time Magazine* entitled "Regulate the Robots" by Massimo Calabresi — brought forth legal implications of robot cars on city streets.

True autonomously driven cars made their first legitimate appearances in DARPA's 150 mile off-road Grand Challenge back in March 2004. One vehicle — the Red Team's modified Hummer shown in **Figure 5** (developed by Carnegie Mellon's Red Whittaker) — managed to drive autonomously a bit over seven of the 150 miles before getting stuck in an embankment. Most vehicles died before even starting or within a mile or so of the starting line.



FIGURE 5. DARPA Grand Challenge 2004 Red Team.

After the race was over, I heard several newscasters lamenting, "If these cars cannot even drive more than seven miles on a protected off-road course in the desert, how can we ever trust robot cars on the roads that we drive each day?"

In 2005, the second Grand Challenge had five of the 23 final entrants complete the entire 132 mile course. The race was won by Stanford University's 'Stanley.' Whittaker's Red Team had two entrants that finished only 20 minutes behind Stanford's six hour and 54 minute winning time. Though shorter than the first race, the 2005 event had mountain passes and tunnels to traverse. Certainly, "robot" cars had come a long way.

Back in 2007, DARPA held their 60 mile Urban Challenge at a closed Air Force base in California. It was set up to represent a real city environment. Tartan Racing's 'Boss' from Carnegie Mellon won the race in four hours and 10 minutes, with Stanford's 'Junior' close behind at four hours and 29 minutes. Virginia Tech's 'Victor Tango' came in third at four hours and 36 minutes. In just three years' time, autonomous driving had developed the navigation software, sensors, and other various visual systems to successfully navigate within an urban environment with other vehicles close by.

Were the naysayers happy? Of course not. There are still many who fear such technology. Those of us in robotics cheered for these accomplishments.

Black Boxes in Automobiles

Another contentious subject relates to the black boxes being installed in modern cars that can record and ultimately reveal a human driver's actions within the vehicle, just before an accident. These are not like the more sophisticated devices required by the FAA in airliners, but are a bit more simplistic. The information gleaned from the device's memory can tell police and investigators just how fast the driver was going, whether or not a seat belt was being worn, when the brake was applied, and other crash details.

It's no big deal for good and/or accident-free drivers, but detractors fear the technology will just escalate into more 'big brother' scenarios. (Maybe one day, we'll have a court session where the black box in a human's car will testify against an autonomous automobile's computer in an accident case.)

Legal Implications

So, I've covered different types of robots that might have legal implications in their use — with or without humans. Drones seem to be the main target of citizen's rights groups at the moment. These groups have cited 175 children and 535 civilians among the 2,348 'other'

people killed by drones in Pakistan as evidence that UAVs should not be used in the Middle East war.

Were these believed innocent people killed because the UAVs were defective? Were they killed because the remote operators were not sufficiently trained in the operation of the vehicle? Was the operator just too anxious to use his weapons and disregarded sensibility? Did the on-the-ground laser targeting individual place the laser beam on the wrong vehicle, person, or building? Who knows. War is terrible, and bad things happen.

The same applies to civilian use of a UAV. Humans with less than honorable intentions can violate basic rights. Is this the fault of the robot manufacturers?

Da Vinci Surgical Robot Lawsuit

Let's steer away from flying and driving robots, and look at medical robots that are currently under scrutiny in several recent lawsuits; one in particular has been filed by the law firm, Alonso Krangle LLP. Plaintiffs have reported many alleged side effects (up to and including death) after undergoing surgeries with the Intuitive Surgical da Vinci surgical robot. The following list (taken from the Alonso Krangle website) shows some of the alleged injuries suffered after a da Vinci procedure:

- Surgical burns to arteries or organs.
- Peritonitis (painful and tender inflammation of the lining of the abdomen).
- Sepsis.
- Excessive bleeding.
- Burning of nearby organs, including the intestines.
- Punctured blood vessels, organs, or arteries.
- Burns and/or tears of the intestines.
- Severe bowel injuries.
- Punctured or cut ureters.
- Vaginal cuff dehiscence (reopening of the incision made to remove the uterus and cervix during a hysterectomy).
- Additional surgical procedures following a robot surgery.
- Wrongful death.

I might be the wrong person to talk about surgeries with unsatisfactory results that were performed by the da Vinci surgical robot (see **Figure 6**). I personally underwent a successful prostate surgery via the da Vinci surgical robot, so I truly understand what is involved.

As a prospective patient, I was interviewed by both my surgeon and anesthesiologist as to the potential problems and possible side effects that could occur during and after surgery.



FIGURE 6. Intuitive Surgical da Vinci surgical robot.

Surgery performed on a human being is not a natural act. We are not given zippers and snap-together organs (like LEGO blocks) at birth. To get to the inside of our bodies, a surgeon must use cutting tools, and loss of blood is an almost certainty with any type of surgery.

Burns and tears of the internal organs, punctured blood vessels, urethras or ureters, severe injury to the bowels, excessive bleeding, and even loss of life can result from common surgeries. Death can result from something entirely different such as a heart attack while 'under the knife.'

Were any of the above injuries the direct result of the robot or could they be the fault of the surgeon using the da Vinci to perform the surgery? Did the robot malfunction or was it poorly designed? Was the surgeon poorly trained? Did the surgeon have a 'God complex' and felt he or she was infallible so ignored safe surgical practices?

Of course, these exact same injuries could happen during arthroscopic or traditional open surgeries.

Final Thoughts

I'd like to thank Tom Green, Editor in Chief of *Robotics Business Review* (**roboticsbusinessreview.com**) and author, Emmet Cole who writes the bi-weekly column, *Robots and the Law*.

Though geared for the professional robotics manufacturer and researcher, *Robotics Business Review* is a treasure trove of up-to-date information about the robotics field. It's not cheap, but I feel it's worth it for the information it provides.

As Tom Green stated, the purpose of robots is "to help, not harm." It seems that most legal implications with robots are the possibility of harming or killing a human through its actions, or by violating a human's rights through unethical operations by a human being.

Of course, the practice of law is extremely complicated, and a short article like this one just scratches the surface of possible issues. Readers of this article might assume I am 'pro' drone, pro da Vinci surgical robot, or pro *any* robot, no matter what.

I am 'pro' any well designed and proven product, whether it's a sophisticated robot or that 'better mousetrap' that always seems to be the goal of any inventor. I am certainly in favor of the many advances in the field and science of robotics. The people in robotics that I know personally hold firmly to ethical and moral values. SV

ROBO-LINKS







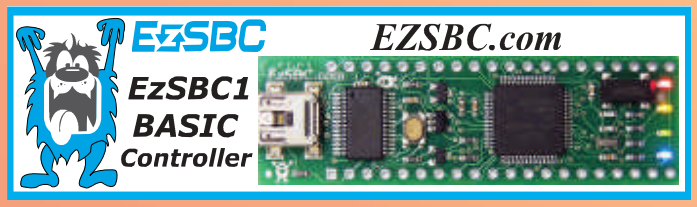


















For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

Advertiser Index

All Electronics Corp. 17, 82
AndyMark 39, 82
AP Circuits 65
Cana Kit Corp. 66, 82
Efficient Computer Systems 57
EZSBC 82
Firgelli 27
Flowol 49
Global Specialties 9
HiTec 2, 82
Machine Design Works 82
Maxbotix 82
PanaVise 49
PCB Pool 39, 82
Pololu Robotics & Electronics ...3, 82
Robotis Back Cover
RobotShop, Inc 61, 82
Robot Power 39
SDP/SI 17
Servo City/Robot Zone 82, 83
Solutions Cubed 19
SuperBrightLEDs 82
Vantec 39
Weirdstuff Warehouse 17

BRAND NEW from ServoCity

Actobotics™

A ROBUST & RE-CONFIGURABLE PRECISION BUILDING SYSTEM

DREAM. DESIGN. BUILD. *REPEAT.*

ACTOBOTICS™ IS... < PRECISE >

Components utilize ball bearings, precision shafting, and tubing - resulting in a superior structure with tight tolerances and low friction.





< INNOVATIVE >

The unique overlapping hole patterns allow for virtually unlimited mounting possibilities.

< COMPATIBLE >

Standardized sizing allows the use of non-Actobotics™ components, including servos, motors, actuators, and R/C vehicle components.



< MOBILE >

Create mechanical motion using gears, sprockets, chains, belts and pulleys.



< ROBUST >

High grade aluminum and stainless steel components provide a rigid foundation.

< PROGRESSIVE >

With hundreds of components already available, our product line continues to expand at a rapid pace.



Easy for both the experienced engineer and novice hobbyist to design and build projects.

< INTUITIVE >

< ACCESSIBLE >

Order everything you need to get your project started, or the one thing you need to finish it.



- Browse thousands of products at **www.ServoCity.com**
- Call **620-221-0123** to place your order by phone
- Contact **sales@servocity.com** for questions

Have a smartphone? Scan this QR code with your barcode scanner and checkout ServoCity on the go!

< CUSTOMER DRIVEN >

Designed with our customers in mind, reducing frustration from component limited projects.



We have the parts for your ideas!



WE SHIP WORLDWIDE!
Order online and receive a real-time international quote.

www.SERVOCITY.com

JOIN OUR ONLINE COMMUNITY!

Copyright 1999 - 2013 Robotzone, LLC. All rights reserved. ServoCity™ and Actobotics™ are registered trademarks of Robotzone, LLC.

ROBOTIS

The Actuator Solution for Full Scaled Robots

DYNAMIXEL PRO



All in One-Actuator

High Power, High Precision

Up to 200W BLDC motor and up to 500,000 pulse per turn (4096 absolute resolution)

Full Modular Solution

Ease of implementation (no additional design for parts needed)
Ease of maintenance (simple part swapping)

Sophisticated Control Algorithms

Position and speed input commands with dual-loop current control

Novel Gear Reduction System

High torque output, light weight, with high impact resistance





Item	Unit	Data		
		H42-20-S300-R	H54-100-S500-R	H54-200-S500-R (Preliminary Product)
Rated voltage	V	24	24	24
No load speed	RPM	28.3	35.2	35
No load current	A	0.61	1.06	1.18
Continuous speed	RPM	15.59	32.7	32.1
Continuous torque	Nm	5.596	21.142	39.131
Continuous current	A	1.989	5.930	9.505
Maximum output power	W	23.64	144.58	262.66
Resolution	Step/turn	304,000	502,000	502,000
Gear ratio	-	304	502	502
Backlash	arcmin	3.5	3.5	3.8
Interface	-	RS-485 / CAN	RS-485 / CAN	RS-485 / CAN
Operating temperature	℃	5~55	5~55	5~55

USA E-mail : america@robotis.com, Tel : +1-949-333-3635
JAPAN E-mail : japan@robotis.com, Tel : +81-3-4330-3660
Etc. E-mail : contactus2@robotis.com, Tel : +82-70-8671-2609
OLLO, BIOLOID, and DYNAMIXEL are registered trademarks of ROBOTIS Co., Ltd.

www.robotis.com